

NASA GRANT REPORT

NASA GRANT NAG8-217

COMPUTERIZED DESIGN OF CONTROLLERS USING DATA MODELS

Prepared by:

Dennis Irwin
Jerrel Mitchell
Enrique Medina
Dan Allwine
Garth Frazier
Mark Duncan

on behalf of:

Department of Electrical and Computer Engineering
Russ College of Engineering and Technology
Ohio University
Athens, Ohio 45701

July 1995

(NASA-CR-198921) COMPUTERIZED
DESIGN OF CONTROLLERS USING DATA
MODELS Final Report (Ohio Univ.)
242 p

N95-30655

Unclass

G3/63 0057748

Final Report

Prepared for:

NASA George C. Marshall Space Flight Center
Marshall Space Flight Center, Alabama 35812

FINAL
10 03 12
57748
p. 2/2

Table of Contents

1	Introduction	1
1.1	Motivations and Initial Objectives	1
1.2	Overview of the Compensator Improvement Program (CIP)	3
1.3	Proposed CIP Updates and Enhancements	5
1.4	Overview of Modern Multivariable Controller Design	6
1.5	Proposed Development Objectives for a Search-Based Modern Multivariable Control System Design Software Package	8
1.6	Overview of Software Testing and Application	9
1.7	Summary of Proposed Development	9
1.8	Other Documentation	10
1.9	Organization of the Report	10
2	Brief Analytical Background	11
2.1	Control System Design Philosophy	12
2.1.1	Classical Control System Design Concepts	13
2.1.2	Modern Multivariable Control System Design Concepts	15
2.2	Search Technique Philosophy	16
3	Compensator Improvement Program	19
3.1	Introduction to CIP	19
3.1.1	Background and Overviews of Original CIP and OUCIP	20
3.2	Conversion of Plane for Discrete-Time Design Improvement	26
3.2.1	Conversion From z-plane to w-plane	26
3.2.2	Conversion from w-plane to z-plane	28
3.3	Improvement of Damping Ratios (ζ)	31
3.3.1	Calculation of Damping Ratios	31
3.3.2	Compensator Damping Ratio Improvement in the s-plane	36
3.3.3	ζ Improvement in the z-plane or w-plane	38
3.3.4	Comparison of s-plane ζ 's and w-plane ζ 's	39
3.4	Compensator DC Gain Improvement	43
3.5	Analysis of Closed Loop Stability	46
3.5.1	Nyquist's Mapping Theorem (Generalized for Multivariable Systems)	46
3.5.2	Using the Nyquist Mapping Theorem in Control System Stability Analysis	47
3.5.3	Discussion of CIP's Implementation of the Nyquist Stability Criterion	49
3.6	Improvement of closed loop disturbance rejection characteristics	50
3.7	Examples and Results	52
3.7.1	Example 1: A Pointing System	52
3.7.2	Pointing System Example (s-plane)	53
3.7.3	Pointing System Example (z-plane)	64

3.7.4	Example 2: Space Shuttle (w-plane)	75
3.8	Conclusions and Recommendations for CIP	94
4	Model and Data Oriented Computer Aided Controller Design System	95
4.1	Control System Design Specifications	96
4.2	Mathematical Programming Principles	100
4.3	Previously Developed Algorithms	103
4.3.1	Simple Approaches	103
4.3.2	The Constraint Improvement Technique	104
4.3.3	The Polak-Mayne Algorithm	107
4.4	Three New Algorithms	110
4.4.1	Algorithm A1	110
4.4.2	Algorithm A2	113
4.4.3	Algorithm A3	117
4.5	Controller Parameterizations	120
4.5.1	State-Space Realizations	120
4.5.2	Rational Function Representations	123
4.6	Partial Derivative Calculations	124
4.6.1	Controller Frequency Response	124
4.6.2	Frequency Response of a Transfer Function	126
4.6.3	Frequency Dependent Singular Values	128
4.6.4	Operator Norms	130
4.6.5	Damping Ratios of Poles and Zeros	131
4.6.6	Eigenvalues	133
4.6.7	Generalized Eigenvalues	134
4.7	Techniques for Obtaining an Initial Controller	135
4.8	Successful Applications	136
4.8.1	The ACES Facility	136
4.8.2	The Hubble Space Telescope	144
4.9	A Performance Comparison	154
4.9.1	The Design Problem	154
4.9.2	Application of the Polak-Mayne Algorithm	158
4.9.3	Application of Algorithm A3	160
4.10	Conclusions and Directions for Further Research for MADCADS	163
5	Conclusions	165
5.1	Summary of Results	165
5.2	Satisfaction of Objectives	165
5.2.1	Completion of CIP Objectives	166
5.2.2	Completion of Search-Based Modern Multivariable Control System Design Software Package Objectives	166
5.2.3	Completion of Software Testing and Application Objectives	167
5.3	Future Work	168

6 References	169
Appendix A: Multivariable Taylor Series	173
Appendix B: Cauchy-Riemann Equations	174
Appendix C: Nomenclature	175
Appendix D: Brief Description of the Software Packages	176
Appendix E: OUCIP User's Guide	E0
Appendix F: MADCADS User's Guide	F0

1 Introduction

This report documents the research and development effort performed by Ohio University under NASA Grant NAG8-217 entitled Computerized Design of Controllers Using Data Models. The originally proposed objectives and tasks are included in sub-sections 1.1 through 1.7. Although several of these objectives and tasks were not met due to a lack of full funding, usable versions of the proposed computer codes were developed and installed on NASA computers. The use of these software packages is described in two standalone user guides included at the end of this document as appendices.

1.1 Motivations and Initial Objectives

The performance objectives in the design of controllers for flexible structures (FS) include vibration suppression, disturbance rejection, and attitude control. FS's are characterized by having many low frequency structural modes that are lightly damped and closely spaced in frequency. In order for controller designs to meet specifications, it is often necessary to incorporate several structural modes within the control system bandwidth. Because of their very low damping, these modes can cause sustained vibrations once excited, and they provide paths of propagation between the disturbances and quantities being controlled and/or regulated. The controller design process must either dampen or suppress (notch) these modes.

Because the modes within the control bandwidth are closely spaced in frequency, the design process, e.g., LQG, H^∞ , loop-at-a-time, μ -synthesis, etc., used to dampen and/or suppress these modes often produces controllers with lightly damped characteristics. This produces significant robustness problems in the presence of model inaccuracy. Experience has shown that models developed either from physical laws or finite element methods (FEM's) do not provide sufficient accuracy for controller designs for FS's with stringent vibration/disturbance/attitude performance specifications. Significant breakthroughs in control system model development, from either physical laws or FEM's, are not expected in the next decade.

The alternative is to develop control system design models from test results. The usual approach is to fabricate the FS, perform testing, and extract an analytical control system design model from the test data. The last step, which is called system identification (ID), is not trivial and is greatly complicated by the FS's being inherently multiple-input, multiple-output (MIMO) in nature. In fact, system ID for FS's is still more of an art than a science and is time consuming and numerically intensive. Furthermore, for the MIMO case the order of the resulting model of the system can easily exceed one hundred. Numerical algorithms used in conventional modern control design to calculate controller parameters are unreliable for problems of this size. To circumvent the order problem, the model is usually reduced using various model reduction schemes. All model reduction schemes are order truncation approaches and, depending on the truncation criterion, can produce models with

different modes and mode shapes than were present in the high order model developed via system ID. A controller design based on the reduced model may or may not produce a closed loop system that satisfies design specifications. If the design does not meet specifications, the designer must either find a better model or fine-tune the design. Unfortunately, it may not be possible to find a better model of the necessary order. Also, fine-tuning the controller with an inaccurate model can yield only limited improvement.

Alternate approaches are obviously needed. A type of approach that can circumvent the pitfalls of the system ID/model reduction/controller design process is to directly utilize data models. The idea is to design controllers and/or fine-tune reduced order controllers by using experimental data instead of a mathematical model of the plant. Test data or frequency response data obtained by applying FFT-based spectrum estimation procedures to test data can be used for this purpose.

The philosophy of designing controllers using data models is not new. One of the most successful ventures in the development of an automated approach to the design of controllers for complex aerospace vehicles using frequency response data models was the Compensator Improvement Program (CIP). CIP was developed for NASA Marshall Space Flight Center in the 1970's to aid in the design of controllers for the ascent flight control systems of the Saturn V and the Space Shuttle (Mitchell, 1973, 1977). Recently the applicability of CIP to the utilization of data models was demonstrated on the NASA Single Structure Control (SSC) Facility at Marshall Space Flight Center. Even more recently the extension of the CIP philosophy to encompass modern multivariable frequency response design criteria was demonstrated by designing controllers for the SSC, again using data models.

With the lack of accurate theoretically derived and/or FEM models, the design of controllers with data models for FS's is the best alternative. The CIP approach of iteratively improving controller designs by molding frequency responses through controller parameter perturbations, provides a sound algorithmic philosophy. As a consequence, it was proposed at the beginning of the project documented in this report to

- (1) enhance, advance and update the existing CIP to handle controller designs for FS's and to operate in a user friendly workstation environment;
- (2) complete the extension of the CIP philosophy and algorithmic approach to simultaneously handle modern multivariable design criteria and the single loop criteria of CIP;
- (3) demonstrate the utility of data model based design on FS control problems of current interest to NASA, e.g., the Hubble Space Telescope and others.

The accomplishment of proposed task (1) above produced a modified version of CIP named OUCIP. The result of task (2) is a software package called Model and Data-Oriented Computer-Aided Design System (MADCADS), which provides for several singular value frequency response shaping constraints, individual input/output pair magnitude response constraints, controller pole and zero damping ratio constraints, and controller pole damping factor constraints. Both OUCIP and MADCADS operate in a user-friendly, workstation environment. Task (3) was partially addressed by using MADCADS to enhance a pointing control system design for the Hubble Space Telescope (Irwin et. al., 1995).

1.2 Overview of the Compensator Improvement Program (CIP)

The Compensator Improvement Program (CIP) is a viable candidate for improving and/or augmenting control system designs for FS's. It can be used to recover lost performance caused by spillover in state-space and/or transfer function (pole placement) designs or to fine-tune loop-at-the-time designs. The essence of CIP is to start with an initial stabilizing design and iteratively increment the design parameters so as to improve broken loop performance measures. The development of CIP has a heritage that started in the Saturn V era and continued into the Space Shuttle era (Mitchell, 1973, 1977). The initial version of CIP was developed to improve designs of controllers for single-input, multiple output systems (Irwin and Mitchell, 1991). Later CIP was extended to handle true multiple input, multiple output systems (Mitchell et. al., 1977).

CIP views the connection of the controller/plant as a multiple loop system. The design philosophy implemented is to iteratively increment the parameters of the controller so that simultaneous improvement of the frequency responses of selected broken loops occur. A broken loop frequency response is the scalar frequency response obtained when each feedback loop is individually opened between the compensator and the plant while all other loops are kept closed. In other words, the i^{TH} broken loop frequency response would be obtained by opening the i^{TH} line between the compensator and the plant and measuring the frequency response from the i^{TH} plant input to the i^{TH} compensator output. (For complex systems, such as FS's, an analysis of this nature is pragmatically impossible by manual design techniques).

Features and characteristics of the original CIP are described as follows:

- (1) The plant or system is assumed to be described in the form of a transfer function matrix. As a model for the plant, CIP requires calculated or experimental frequency response data for each element of this matrix. By using frequency response data to model the plant, numerical problems in handling large order systems are eliminated, and experimentally determined frequency responses can be directly accommodated by CIP.

- (2) Performance specifications can be made frequency dependent without necessarily increasing the controller order. This allows the user to independently specify constraints for both the gain and phase stabilization regions.
- (3) The controller is described as a transfer function matrix in which each element is represented as a ratio of first and second order factors. For continuous-time controllers these are s-plane functions, whereas for digital controllers these are w-plane functions. The coefficients of these factors are varied by CIP to improve the system performance. By not varying certain coefficients, CIP can place restrictions on a controller element. In particular, the D.C. gain of an element can be held constant to assure steady-state error performance, the coefficients of first order factors can be constrained to be positive in order to avoid first order right half plane poles and/or zeros, or the damping ratios of second order factors can be specified to be above minimum values in order to assure robustness of the controller.
- (4) CIP tests for system stability on each iteration.
- (5) The coefficient change vector computed by CIP assures that from iteration to iteration an improved design results in the sense that no performance measurement is degraded.

The original CIP algorithm begins with the specification of frequency response data for each element of the plant transfer function matrix, the initial compensation matrix, the desired specifications, etc. The main iterative procedure is then entered. In this procedure, the performance measurements of the system are evaluated by opening each feedback loop, with all other loops closed, and determining stability and attenuation margins (called broken loop analysis). These performance measurements are compared to the design specifications; if all specifications are satisfied, the design is complete, and the process is terminated. Otherwise, performance measurements satisfying design specifications are discarded, leaving a list of active constraints (or items that need to be improved). The convergence of the iterative process is checked by assuring that the improvement in performance is greater than a user specified value. Next, gradient vectors of unsatisfied performance measurements with respect to the free parameters (coefficients) of the compensation matrix are computed. The gradient vectors are used by the Constraint Improvement Technique (CIT) to compute the compensator coefficient change vector (Mitchell, 1972). The change vector is used to increment the compensation matrix so that an improved solution is assured. Then, the iterative process is repeated.

An interesting property of the change vector computed by CIT is that a positive inner-product with all gradient vectors can be assured; as a consequence, it is theoretically possible to simultaneously improve all unsatisfied performance measurements. However, from a practical point of view, zero improvements or small

degradations in some performance measurements can be allowed in order to obtain large improvements in one or more of the others. More important is the implication that large degradations in any satisfied performance measurements are avoided by CIT. CIT is the cornerstone of the Compensator Improvement Program.

1.3 Proposed CIP Updates and Enhancements

The original CIP was developed to improve control system designs for complex ascent flight control systems. It was anticipated at the proposal stage of the work documented here that with some modest updates and enhancements CIP could become a valuable tool for designing/fine-tuning of controllers for FS's. Proposed tasks for enhancing CIP for this purpose were as follows:

- (1) Perform modifications so either digital z-domain or continuous s-domain controllers could be produced. CIP was limited to continuous s-plane or digital w-plane controller designs.
- (2) Provide the option for the inclusion of vibration suppression and disturbance/noise rejection specifications. The existing form of CIP could only handle specifications related to phase and gain stabilization.
- (3) Provide the option of independent frequency response specifications for each loop. Because of hardware limitations many loops must be designed with different bandwidths; hence, the desired design specifications can differ.
- (4) Include the option of specifying the controller in a state-space format. If state space designs are to be improved, then CIP should be able to directly handle the state space format rather than require conversion to transfer function matrix format.
- (5) Modify so that closed loop specifications could be made. The existing CIP only attempted to achieve open loop specifications, viz., gain margins, phase margins, attenuation margins, and/or stability margins.
- (6) For open loop specifications, provide the option of locating the loop breaking points either before the plant or before the controller.
- (7) Include pre and post-analysis of system singular values. This will provide the designer with additional performance and robustness information.
- (8) Update the CIP code to operate in a workstation environment, i.e., user friendly and interactive, rather than a batch environment.

The core of the existing CIP was solid, well developed, and had passed numerous tests. The core was to be modified only to make the code more maintainable and to include modern, reliable numerical algorithms. This shell approach to the enhancement of CIP allows for a reasonably swift and very reliable implementation of the tasks discussed above.

1.4 Overview of Modern Multivariable Controller Design

As mentioned above, the extension of the CIP philosophy and algorithmic approach to handle modern multivariable design criteria generated the MADCADS software package. This sub-section discusses some motivations for these extensions to the search-based controller redesign approach.

There is currently a great deal of interest in the control community in robust control in general and H^∞ control in particular. Much of this interest has been caused by the discovery of state-space formulas for calculating plant order controllers which satisfy an H^∞ norm constraint on a closed loop transfer function (Glover and Doyle, 1988), (Maciejowski, 1989). These formulas have since been used in conjunction with optimal projection methods to yield reduced order controllers which are optimal in an H_2 sense among controllers of the same order which satisfy the H^∞ norm constraint (Haddad, et. al., 1991). The latter approach is motivated by the desire to directly trade off performance, measured in an H_2 sense, with robustness, measured in an H^∞ sense. The major weakness of this H_2/H^∞ design combination appears to be computational.

At this time, however, analytical techniques are not sufficiently well developed to allow the design of controllers which perform the tradeoff of disturbance attenuation, command tracking and robustness in a way that is completely general. The approach followed in this project is to cast all multivariable performance and robustness criteria in terms of frequency response dependent bounds on the singular values of various open loop and closed loop frequency response matrices. This is precisely the approach taken in conventional frequency weighted H^∞ control formulations, with the exception that in the latter case all such problems, even robustness and command tracking problems, are cast in the form of a disturbance attenuation problem for which solution formulas exist.

The differences between the standard approach to variants of H^∞ control and that originally proposed by Ohio University for this project are:

- (1) The design freedom is directly incorporated into the design process via frequency-dependent singular value constraints, rather than through the introduction of frequency dependent weighting factors. These weighting factors contribute directly to the dynamical order of the resulting controller in conventional H^∞ design methodology.

- (2) The controller order is pre-specified and can be smaller than the plant order. Of course, the order specified must be consistent with the design constraints. This is consistent with the philosophy that controller order should be a parameter over which the designer has choice.
- (3) It is not necessary that the problem of interest be cast as a disturbance attenuation problem. The need to recast the original design problem definition, which is a major source of confusion in conventional robust control, is therefore eliminated.
- (4) The controller structure can be prespecified. That is, the controller can be forced to be diagonal or some other type of canonical form. Thus, the controller can be designed to take advantage of particular hardware architectures.
- (5) The design is carried out via an iterative numerical procedure which allows for a great deal of designer interaction when implemented in a graphical workstation environment.
- (6) Unlike conventional robust control approaches, the technique used here does not require an analytical design model. Experimental frequency responses are sufficient to effect controller designs.
- (7) Microscopic (individual frequency response matrix element) as well as macroscopic (norm bounds on frequency response matrices) design constraints can be handled simultaneously.

The technique used here formulates the design problem as a strict constraint problem rather than as an optimization problem. Since these constraints are in terms of the frequency dependent singular values of transfer function matrices, these infinite dimensional constraints must be converted to a large (but finite) number of constraints. The approach is similar to that of Boyd (1988). However, the present approach does not use the affine parameterization of stabilizing controllers together with a specialized controller architecture in order to insure that the resulting mathematical programming problem is convex. Rather, CIT-like techniques are utilized to achieve the design constraints. The advantage is mainly one of resulting controller order although, since these methods only deal with active constraints, the computational load is likely to be relatively light.

One of the main advantages of the search-based approach to multivariable controller design and/or fine-tuning presented here lies in the fact that any constraint for which analytical gradients can be calculated can be incorporated into the design algorithm. At the beginning of this project, the following were considered as possible design criteria and/or features of the approach:

- (1) Singular value robustness criteria (macroscopic)
- (2) Singular value disturbance rejection criteria (macroscopic)
- (3) Singular value command tracking criteria (macroscopic)
- (4) Constraints on controller eigenvalue locations (microscopic)
- (5) Constraints on controller transmission zero locations (microscopic)
- (6) Constraints on individual transfer function matrix elements (microscopic). This can allow for differing bandwidths in dominant input/output channels.
- (7) Decoupling constraints (microscopic and macroscopic).
- (8) Controller order and structure (microscopic).
- (9) Frequency response data as a plant model, effectively eliminating the need for an analytical model.

1.5 Proposed Development Objectives for a Search-Based Modern Multivariable Control System Design Software Package

The full realization of the search-based approach to multivariable control design requires two major efforts. The first is the investigation of the simultaneous implementation of microscopic and macroscopic design constraints. The other is the realization of the algorithm in a fully interactive and graphical design workstation environment. The originally proposed tasks for accomplishing these major components of a practical design system were:

- (1) The incorporation of a full complement of modern multivariable performance and robustness criteria into the code existing at the time of the proposal, which included only those criteria necessary for proof-of-concept studies.
- (2) The incorporation of damping ratio constraints.
- (3) The incorporation of single input/output pair transmission constraints.
- (4) The investigation and implementation of the most desirable state-space structures to realize a particular set of design constraints. Preliminary work had indicated that the structure of the controller realization could affect the rate of convergence of the search algorithm. The complexity of the gradient calculation step of the search algorithm was also known to be affected by the controller structure.

- (5) The implementation of the design software in a professional workstation environment.
- (6) The development of effective graphical algorithm evaluation aids to allow effective designer interaction. The limited experience with the algorithm at the time of the proposal had indicated that algorithm convergence was often enhanced by prudent user interaction.

1.6 Overview of Software Testing and Application

It is a fact that the evaluation of controller design methodologies is best done by application to realistic design problems. It was proposed that the effectiveness of the CIP enhancements and multivariable controller design algorithm development would be monitored via their application to problems of immediate interest to MSFC personnel. In most cases, the controller design would be performed using experimentally derived frequency response data. Proposed tasks included:

- (1) The application of the enhanced CIP to the problem of vibration suppression for the Hubble Space Telescope (HST) solar power panels.
- (2) The application of the modern multivariable search-based design algorithms to the HST vibration suppression problem.
- (3) The application of the multivariable algorithms to the CASES ground facility.
- (4) The application of enhanced CIP to the Single Structure Control facility.

1.7 Summary of Proposed Development

As mentioned above, it was proposed to advance the state-of-the-art of controller design using data models by (1) enhancing and augmenting CIP, (2) completing the development of search-based approach to achieve modern frequency response controller designs and (3) demonstrating the utility of the resulting comprehensive computerized controller design methodology to hardware problems of interest to MSFC personnel. In the design of controllers for FS's there are two basic types of design specifications, microscopic and macroscopic. The design specifications for CIP are microscopic in nature, e.g., phase margins, gain margins, stability margins, and attenuation margins for specific loops. On the other hand, the specifications of modern frequency response design approaches are macroscopic in nature, e.g., infinity norm or singular value based measures of system robustness, disturbance rejection, etc. Clearly the achievement of both types of specifications are desirable. Microscopic specifications can set minimum dynamic and/or stability standards for individual loops whose bandwidths may have to be significantly different due to hardware and/or physical limitations, whereas macroscopic specifications set minimum

combined standards of all the feedback loops working together to maximize robustness and/or minimize the effects of disturbances. An ideal FS's controller design tool should be able to handle specifications of both types -- the original goal of this research.

1.8 Other Documentation

Additional information regarding the software that was developed or modified during the course of this project is available in the form of two user guides, included as appendices to this report:

- (1) OUCIP: Ohio University Compensator Improvement Program, User's Guide
- (2) MADCADS: Model and Data-Oriented Computer-Aided Design System, User's Guide

Much of this report is based on the Master of Science thesis of Mark Duncan (Duncan, 1994) and the Ph.D. dissertation of W. Garth Frazier (Frazier, 1993), which contain some supplemental material that is not included in this document.

1.9 Organization of the Report

The remainder of this report is organized as follows. Section 2 contains a brief discussion of the general controller design philosophy for both the classical and modern approaches and some background material about the application of search techniques to the controller redesign problem. The Compensator Improvement Program is covered in section 3. Specifically, the original work and new developments are described and the enhanced technique is applied to several example problems. Section 4 continues with a discussion of the theory and development behind MADCADS and also includes the results from application to two real-world problems. Section 5 offers some conclusions about the project and makes recommendations for future work. Section 6 includes a list of references. Appendices A through C contain some mathematical proofs and support information, while Appendix D contains a brief description of the software packages. Appendices E and F are the user guides for OUCIP and MADCADS, respectively.

2 Brief Analytical Background

Numerous methods have been developed and employed over the years for designing controllers for feedback control systems. For the very important class of linear, time-invariant (LTI) systems, these methods range from the well known graphical procedures for single-input, single-output (SISO) design that use well-known tools such as root-locus, Bode plots, and Nyquist plots, through MIMO extensions of these classical concepts, e.g., sequential loop closing (one-controller-at-a-time or 1-CAT) (Mitchell, 1984), characteristic locus, and Nyquist Array methods, to the more recently developed linear-quadratic-gaussian optimization with loop transfer recovery (LQG/LTR) procedure and H-infinity (H_∞) optimization controller synthesis (Maciejowski, 1989, chaps. 5 and 6). Each of these methods has particular advantages and disadvantages; for example, controller design using graphical procedures has the advantage that it usually results in controllers that are low in order (complexity) relative to that of the open-loop system or *plant*. Another advantage is that designs can be achieved by using a variety of plant models, such as a nonparametric model obtained from experimental data (a data model), a parametric model obtained from experimental data (an identified model), or a parametric model based upon physical principles (an analytical model). These procedures have the serious disadvantage that achieving multiple design specifications can be extremely difficult, especially in the case of complex, multiple-input, multiple-output (MIMO) systems.

The analytical synthesis design methods, such as LQG/LTR and H_∞ -optimization, have the advantage that they are well suited to design controllers for SISO and MIMO plants. They have the disadvantages that a nonparametric model of the plant cannot be used, the controller order is generically greater than the order of the plant model, and the design constraints that are actually desired must usually be specified implicitly by choosing various parameters and weighting functions. More importantly, the variety and number of design constraints that these methods can simultaneously encompass is quite limited. Another potential drawback is that the resulting controller often cancels lightly damped poles and zeros of the plant. In some instances this may not be critical, but in others instances it can lead to unpredictable results upon actual implementation. If these lightly damped dynamics are modeled inaccurately, or if they change under different operating conditions, closed-loop stability can be lost, or serious performance degradations can occur. Unfortunately, it is usually difficult (but not impossible) to prevent these cancellations without sacrificing other design objectives.

Another approach to controller design is to use parameter search methods to systematically change the free parameters of a nominal controller to achieve multiple design constraints. Several researchers including Mitchell (1973), Zakian and Al-Naib (1973), Polak and Mayne (1976), Edmunds (1979), Kreisselmeier and Steinhauser (1979), Boyd, et. al. (1988), and Frazier and Irwin (1993) have developed such methods, which are based upon the principles of mathematical programming. A

recent perspective on search-based methods for controller design is given by Ng (1993). Several of these methods have the capability of using parametric and nonparametric plant models and can encompass a wide variety and a large number of design constraints simultaneously. These two features are the advantages that distinguish search-based methods from analytical synthesis methods. The primary disadvantages are that the control system designer usually needs to provide an initial, stabilizing controller and it is virtually impossible in most instances to obtain sufficient conditions on the design specifications that guarantee that the algorithm will achieve a successful design or even to determine the existence an acceptable design for a given set of specifications. Experience has shown, however, that with a "good" initial controller and well conceived design objectives, satisfying the constraints (design objectives) is often possible.

The nature of controller design makes it an ideal application for search-based methods. For example, typical control system design specifications, such as achieving desired levels of disturbance rejection, noise attenuation, and stability robustness to plant variations, while limiting control effort, can be cast as a set of inequality constraints on functions of the free parameters of a controller. Nevertheless, research in this area of controller design appears to have decreased in recent years, perhaps in part because of the mathematical elegance and apparent power of the modern analytical methods. It is felt, however, that with the ever increasing rate of improvement of computer technology, especially in graphical user interfaces and floating-point processors, that search-based methods can certainly be used in conjunction with analytical methods if not as an alternative. Therefore, research efforts in this area are promising. This approach to controller design is the subject of this research effort.

Two fundamental steps are required in order to devise an algorithm that implements the application of search techniques to the problem of controller design and/or fine tuning. First, a suitable set of design specifications must be determined and translated into a set of usable constraints. Second, a search algorithm must be found that can modify the controller parameters until these constraints are satisfied. These two stages are definitely interdependent; the selection of constraints will depend on the solution method to be used, and vice-versa. The following sub-sections discuss the basic concepts of control design problem formulation and search algorithms that are used throughout the rest of this development.

2.1 Control System Design Philosophy

Before a search technique can be applied, a control system design problem must be defined. The nature of this definition will be determined by the control philosophy used. Numerous techniques exist for the formulation of a control design problem. In this work, two main approaches are followed. In the case of CIP, classical

techniques are used. For MADCADS, modern multivariable concepts are implemented. The next two sub-sections discuss the details of these two approaches.

2.1.1 Classical Control System Design Concepts

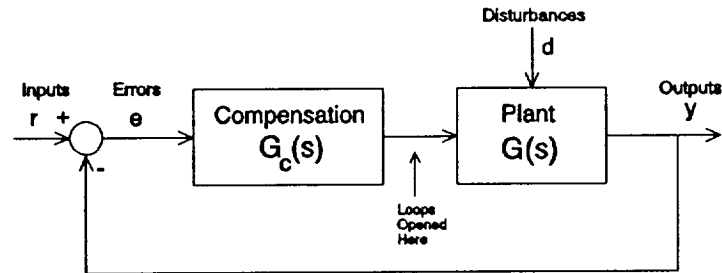


Figure 2.1 Block Diagram of a Multivariable Feedback Control System

A multi-input, multi-output feedback system is depicted in Figure 2.1. A classical approach for designing the compensator $G_c(s)$ relies on the premise that desired closed loop performance characteristics can be indirectly achieved by satisfying open loop frequency response specifications. This concept is obviously an extension of the single-input, single-output case. The idea is to open one loop at a time between the controller and the plant and compute, at each opening, certain functions of the frequency response that will determine closed loop performance. Each of the frequency responses obtained by these loop breakings is called a *broken loop frequency response*.

The open loop or broken loop design philosophy is to phase stabilize modes in the control system bandwidth, i.e., design the open loop controller for each loop so that no mode results in an encirclement of the $-1+j0$ point on a polar frequency response plot, and in fact maintains a specified distance from the $-1+j0$ point. In this region most of the broken loop frequency response is above zero dB (above unity gain). The gain stabilization region corresponds to those frequencies above the control system bandwidth. The design philosophy in the gain stabilization region is to attenuate the modes as much as possible to minimize their impact on closed loop performance. In the closed loop, modes that are phase stabilized can have their natural frequencies and damping ratios change considerably from the broken loop values, whereas closed loop modes that have been gain stabilized will have natural frequencies and damping ratios near their broken loop values. The broken loop quantities that are related to closed loop performance are:

Gain Margins

A gain margin is a measure of the distance from the $-1+j0$ point to a $\pm 180^\circ$ crossing of a broken loop frequency response. The classical definition is as follows: a gain margin is defined as the inverse of the magnitude at the

frequency for which the broken loop frequency response crosses the negative real axis in the complex plane. The classical definition provides the amount of pure gain change that must be made to produce instability. Of course this assumes the closed loop system is stable.

Phase Margins

A phase margin is defined as the amount of phase that must be added or subtracted at a frequency for which the broken loop frequency response magnitude is unity to produce instability. If the broken loop phase at the unity magnitude point lies in the interval $[0, -180^\circ]$, the phase margin must be subtracted; otherwise it should be added.

Stability Margins

A stability margin is defined as a closest approach of a broken loop frequency response curve to the $-1 + j0$ point on a polar plot.

Attenuation Margins

In this context, an attenuation margin is defined as a peak value of the broken loop frequency response magnitude.

Gain margins, phase margins, and/or stability margins design specifications are used to assure closed loop damping and to some extent robustness. These specifications are usually made in the phase stabilization region. Attenuation margins provide a measure of modal attenuation in the gain stabilization region.

Direct improvement of closed loop performance is desirable. For FS's it is well known that many performance objectives can be met by designing controllers with lightly damped characteristics. Designs of this nature typically intermingle lightly damped controller poles and zeros with lightly damped poles and zeros of the plant model. The consequence is a problem of design robustness with respect to model uncertainties. This problem is lessened by limiting the damping factors of the controller poles and zeros.

Another design requirement of FS's is to reduce the propagation of disturbances, D , to the system outputs, Y ; this is called designing for disturbance rejection. Closed loop disturbance rejection can be accomplished by imposing constraints on the rms and peak values of the closed loop frequency response magnitudes from disturbances D to outputs Y . Many FS's control systems are required to be steady-state following systems, i.e., the respective outputs are required to follow respective inputs when the inputs are standard inputs such as steps, ramps, or parabolas. This is normally referred to as designing to satisfy steady-state error requirements. Desired closed loop steady-state error characteristics are obtained by including an appropriate number of pure integrations in error signal broken loops and by placing constraints on the D.C. gains of selected compensator elements, viz., requiring DC gains of certain compensator elements to be above minimum values.

2.1.2 Modern Multivariable Control System Design Concepts

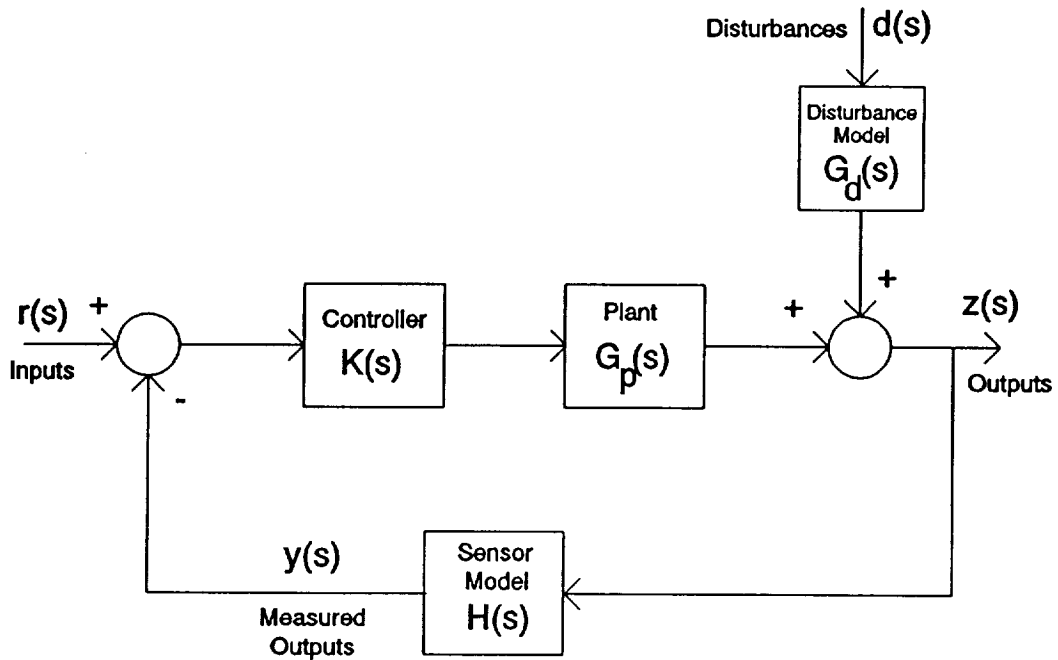


Figure 2.2 General Block Diagram for Multivariable Design

A basic multivariable feedback control system is shown in Figure 2.2. In a modern controller design setting, the performance of the system is commonly measured by studying the following quantities:

$F_o(s) = I + G_p(s) K(s) H(s)$: Return difference matrix evaluated at the plant output

$F_i(s) = I + K(s) H(s) G_p(s)$: Return difference matrix evaluated at the plant input

$S_o(s) = F_o^{-1}(s)$: Output sensitivity function

$S_i(s) = F_i^{-1}(s)$: Input Sensitivity function

$T_o(s) = S_o(s) G_p(s) K(s)$: Output complementary sensitivity function

For example, the system outputs due to disturbances can be written as

$$z(s) = S_o(s) G_d(s) d(s) , \quad (2.1)$$

which yields the disturbance to output transfer function matrix

$$S_o(s) G_d(s) = [I + G_p(s) K(s) H(s)]^{-1} G_d(s) . \quad (2.2)$$

Thus the disturbance rejection capabilities of the system will be "good" if the transfer function matrix in Equation 2.2 is "small". Since the system is multivariable in nature, the "size" of the transfer function matrix is measured in terms of its singular values. Hence, "good" disturbance rejection capabilities over a specified frequency range can be obtained by forcing the maximum singular value of the transfer function matrix in Equation 2.2 to be "small" for the specified frequencies.

Similarly, the maximum singular value frequency response from $r(s)$ to $z(s)$ represented by

$$\sigma_{\max}[T_o(s)] = \sigma_{\max}\left[[I + G_p(s) K(s) H(s)]^{-1} G_p(s) K(s) \right] \quad (2.3)$$

could be forced to be "near" unity over a particular frequency range in order to maintain "good" command tracking capability.

Other constraints can be defined to design for such characteristics as control system robustness to uncertainties in the plant model or to produce desired stability margins.

2.2 Search Technique Philosophy

The first candidate technique for solving the problem of having multiple control design objectives that depend on a set of compensator parameters would be a multiple objective optimization technique. A multiple objective optimization problem can be posed as follows.

It is desired to determine values for the elements of the parameter vector $\mathbf{x} = [x_1 \ x_2 \ \dots \ x_n]^T$ in order to maximize (minimize) the objective functions

$$f_1(\mathbf{x}), f_2(\mathbf{x}), \dots, f_m(\mathbf{x})$$

subject to the constraints

$$f_{m+1}(\mathbf{x}) \geq a_{m+1}, f_{m+2}(\mathbf{x}) \geq a_{m+2}, \dots, f_{m+p}(\mathbf{x}) \geq a_{m+p}.$$

In general, it is impossible to maximize (minimize) multiple functions with the same parameter vector \mathbf{x} . (An obvious case for which a solution might exist is when $f_1 \dots f_m$ are functions of different variables. However, even in this case a solution may not exist.) A compromise solution is usually the best option. The standard approach to obtaining a compromise solution is to solve the modified problem

$$\text{maximize (minimize) } \sum_{i=1}^m c_i f_i(\mathbf{x})$$

subject to the constraints

$$f_{m+1}(\mathbf{x}) \geq a_{m+1}, f_{m+2}(\mathbf{x}) \geq a_{m+2}, \dots, f_{m+p}(\mathbf{x}) \geq a_{m+p},$$

where \mathbf{x} is defined as before.

A problem with this standard compromise solution approach is that it will simply yield a solution at or near the minimum of a dominant function in the sum, if a minimum occurs in the feasible region.

A more realistic problem formulation, and one that lends itself to computer implementation, is to define the problem as one of multiple constraints that have to be satisfied, without defining explicit objective functions that must be maximized (minimized). In other words, it is desired to

determine \mathbf{x} such that

$$f_1(\mathbf{x}) \geq a_1, f_2(\mathbf{x}) \geq a_2, \dots, f_m(\mathbf{x}) \geq a_m,$$

and

$$f_{m+1}(\mathbf{x}) \geq a_{m+1}, f_{m+2}(\mathbf{x}) \geq a_{m+2}, \dots, f_{m+p}(\mathbf{x}) \geq a_{m+p}.$$

The fundamental goal of a method for solving this problem is to find a point $\hat{\mathbf{x}}$ in the parameter space for which all constraints are satisfied, if such a point exists. If such a solution does not exist, the method should yield a solution that is a best compromise. A practical method for solving this problem is to iteratively adjust the parameter vector \mathbf{x} in a controlled fashion until all constraints are satisfied (or no further improvement can be achieved). The Constraint Improvement Technique (CIT), developed by Mitchell (1972), implements this type of solution approach.

The basic philosophy of CIT can be described as follows:

at the i^{th} iteration

find a step length k and a directional vector d

such that for all $j \in \{1, 2, \dots, m+p\}$ for which

$f_j(x) < a_j$, (i.e., the j^{th} constraint is unsatisfied)

it is true that

$f_j(x_{i+1}) > f_j(x_i)$, (i.e., the j^{th} constraint is improved)

where $x_{i+1} = x_i + kd$.

Two issues are of importance in this method. First, a means of computing a suitable directional vector d is needed. Second, in the ideal situation, the step length k is computed in such a way that all violated constraints are improved, i.e.,

$$f_j(x_{i+1}) > f_j(x_i) \quad , \quad j=1,2,\dots,q \quad , \quad (2.4)$$

where q is the number of violated constraints at the i^{th} iteration. However, since it is not always the case that all violated constraints can be simultaneously improved, a value of k might have to be chosen that will satisfy

$$\sum_{j=1}^q [f_j(x_{i+1}) - f_j(x_i)] > 0 \quad , \quad (2.5)$$

i.e., the sum of constraint improvements is greater than the sum of constraint degradations.

An in depth discussion of the characteristics of search techniques as applied to the problem at hand are given in sub-sections 4.2 to 4.4.

3 Compensator Improvement Program

This section describes the Compensator Improvement Program (CIP) in detail. As an introduction, the general ideas behind the program are discussed and a brief history of the development of CIP is given. The additions to CIP developed as part of the work documented in this report are studied in the rest of the section.

The CIP documentation in this section is organized in seven sub-sections. Sub-section 3.1 is a background discussion to controller design and CIP. Sub-section 3.2 discusses the method in which z-plane data is converted to the w-plane for use by CIP. Sub-section 3.3 contains the theory behind the improvement of the compensator damping ratios. The method implemented for improving the compensator DC gains is examined in Sub-section 3.4. Section 3.5 is devoted to Multi-Input/Multi-Output (MIMO) system closed loop stability using the generalized Nyquist stability criterion. The procedure followed in the addition of closed loop disturbance rejection specifications to CIP is discussed in Sub-section 3.6. Results of the application of OUCIP to several examples are presented in Sub-section 3.7. Sub-section 3.8 contains some conclusions about the development of OUCIP and results, as well as shortcomings of the current version of OUCIP and suggestions for future work.

3.1 Introduction to CIP

Modern control system design has become very complex, trying to meet the needs of today's industry, space program and consumer. Many times analytical design techniques fail to meet the requirements that this new era of technology demands. Systems are often difficult or impossible to model analytically, and even when possible, the small number of design specifications that can be achieved hardly makes the design task worth while. Analytical design techniques often result in controllers of very high orders which can be expensive and/or impossible to implement in hardware. An alternative to analytical controller design techniques is numerical based techniques in which a data model of a plant is used, and the parameters of a fixed order controller are varied to achieve design specifications. This is the general idea behind the Compensator Improvement Program (CIP) (J.R. Mitchell, et. al., 1977).

CIP was developed in the 1970's for use in improving open loop frequency response specifications of Multi-Input/Multi-Output (MIMO) systems. It uses a directional search technique to iteratively modify the characteristics of an initial compensator in such a way that several design specifications of the system are iteratively improved.

The original CIP of the 1970's was executed as a batch file, and the user had no way of interacting with the progress of the design improvement. Today's CIP, called OUCIP, has a graphical user interface in which the user can pause the execution and manipulate the design specifications, turn specifications on or off, and select graphical

outputs. Also in this new format, the user can save an application at any iteration, for execution at a later time. Real time plotting of the frequency responses of many systems is also available, including open loop plant, compensator, compensated open loop system, closed loop system, and determinant of the return difference matrix, which is used to determine closed loop stability.

The original versions of CIP could not directly improve z-plane designs. Before executing CIP, the user had to first convert all data to the w-plane. Direct z-plane design improvement was studied and found to be highly sensitive to machine precision. To avoid this problem it was then decided to give CIP the capability to automatically convert all appropriate data to the w-plane to preform the design and then convert back to the z-plane once the design improvement was finished.

Another problem with the original CIP was that there was no way to include the compensator DC gains and damping ratios as design specifications. The only control that the user had over these specifications was to hold the DC gains constant or allow them to vary. For the damping ratios, the user could set a minimum value that they could never go below. This minimum value for compensator damping ratios could not be greater than the minimum initial damping ratio of all compensator elements or CIP would terminate with a message that told the user that the initial compensator damping ratios were violated. This posed a problem for initial compensators with low DC gains and/or lightly damped terms. It was determined that a way of improving these specifications to some desired value was needed.

Stability checking in the original CIP was done by using the Nyquist criteria to check the stability of each loop-at-a-time (explained in section 3.1.1) compensated open loop system treating them as Single-Input/Single-Output (SISO) Systems. In order to implement the Nyquist criteria, the number of open loop poles in the right half plane is needed. For a MIMO system being analyzed a loop-at-a-time, this information is not trivial to obtain. In fact it was discovered that the loop-at-a-time compensated open loop systems may go unstable while the closed loop system is still stable. Therefore a better way to check for closed loop stability was needed and, as a consequence, the generalized Nyquist approach was selected and implemented in OUCIP.

The work presented in this section includes several additions and improvements to the original CIP. In particular, there are five areas in which additions have been made; z-plane design improvement, compensator damping ratios, compensator DC gains, closed loop stability, and closed loop disturbance rejection.

3.1.1 Background and Overviews of Original CIP and OUCIP

Figure 3.1 is the general block diagram used in OUCIP. All signal paths are vectors as indicated by double lines. $U(s)$ is the set point vector inputs. $E(s)$ is the

vector error signal between the set points $U(s)$ and the measured outputs $Y(s)$. $Z(s)$ is the physical output vector. $X(s)$, the control inputs, is the point where the loops are assumed to be broken for loop-at-a-time analysis of a system by CIP. $D(s)$ is the vector disturbance inputs to the plant. Although these may be modeled as being after the plant, no generality is lost by the portrayal.

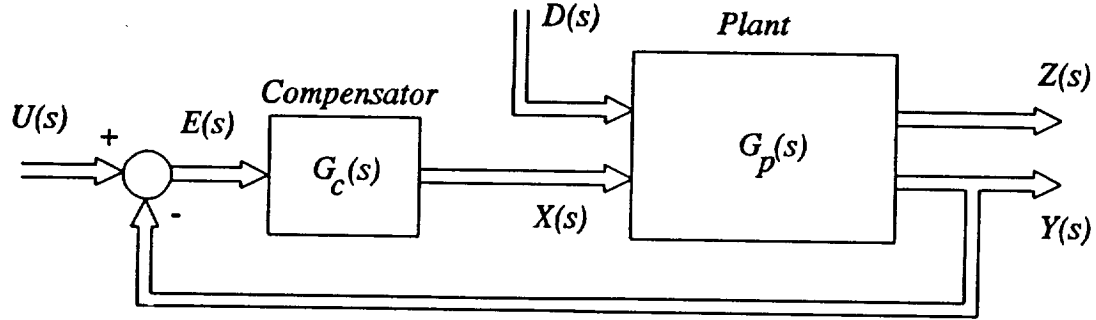


Figure 3.1: General Block Diagram Used in OUCIP

The other blocks shown in Figure 3.1 are $G_c(s)$ and $G_p(s)$. These are the compensation and the plant matrices, respectively. The compensation matrix is given to OUCIP by the user in the form of cascaded first and second order polynomials for each element of the matrix. For example, the ij -th element of the compensation matrix has the general form

$$G_{ij}(s) = (\text{gain}) \frac{\prod_{l=1}^{N1} (ZA_l + ZB_l s) \prod_{l=1}^{N2} (ZC_l + ZD_l s + ZE_l s^2)}{\prod_{l=1}^{M1} (PA_l + PB_l s) \prod_{l=1}^{M2} (PC_l + PD_l s + PE_l s^2)}, \quad (3.1)$$

where $N1$, $N2$, $M1$, and $M2$ are the number of first and second order numerator and denominator factors respectively. The plant matrix is given to OUCIP as frequency response data, taken either experimentally from the plant or formed from a mathematical model of the plant.

Figure 3.2 is a simplified flow chart of the original CIP. In the original versions of CIP, the program was never paused during execution. First, the plant data, compensation data and the design specification data were read from files. CIP then began the iterative loop, where design improvement occurred. CIP calculated the compensated open loop system, also known as a broken loop system. The program then determined the performance measurements which needed improvement - gain margins, phase margins, attenuation levels and stability margins (closest approach of frequency responses to the $-1 + j0$ point). Next, the gradient vector of each unsatisfied performance measurement was calculated and inserted into a respective row of the gradient matrix. If performance could be further improved, the change

vector, commonly known as the directional vector, was evaluated and used to update the compensation such that improved performance is assured. This series of events continued until: (1) all specifications were satisfied or (2) the amount of overall improvement to the system had diminished below a set tolerance level. At this point, the program wrote output into files and the program halted execution.

The above described version of CIP had no method for the user to interact with the iterative process. In addition, there were other design specifications that were missing, eg., compensator damping ratios or DC gains which help to improve robustness, steady state error, and disturbance rejection.

Figure 3.3 is a simplified flow chart of OUCIP. The flow chart includes enhancements both to the numerical capabilities and to the user interface of CIP. Although it is not shown on Figure 3.3, it is assumed that all submenus return to the MAIN MENU upon exiting. The entire MAIN MENU of OUCIP is shown in Figure 3.3, but only the submenus necessary for executing iterations of the search algorithm are shown in the flow graph.

Execution of OUCIP begins by the appearance of the MAIN MENU to the screen. The user can then make many choices before implementing the design improvement section of OUCIP. The most obvious first step is selecting the data with which to work. This is accomplished by the use of the FILE submenu. In the FILE submenu, there is a selection called Retrieve Setup. When this selection is chosen, a window containing all CIP executable data files is opened. The user then selects via the mouse, the desired system, compensation, and specification files. This Retrieve Setup window then closes and OUCIP returns to the main window. The user then has the choice of executing OUCIP with the selected files, or the user can select one of the other menus such as: (1) PARAMETERS, where the step size, title, improvement tolerances and other parameters can be changed. (2) ACTIVATE, where the user can specify which design specifications are turned on or off. These include: (a) Relative Stability, (b) Disturbance Rejection, (c) Compensator Damping Ratios, and (d) Compensator DC gains. (3) GRAPHICS, where the plotting windows can be created for real time display of chosen frequency responses of the system.

There are two options in the submenu EXECUTE. The first is SINGLE, meaning only one iteration of the design improvement is implemented. After this single iteration is finished, OUCIP returns to the main menu. The second option is MULTIPLE. When MULTIPLE is selected, an Execution Control Window is opened. In this window the user types the number of iterations desired for OUCIP to attempt. Also in this window are two buttons, SINGLE and MULTIPLE. Clicking on SINGLE executes one iteration and MULTIPLE executes the number of iterations that the user has requested. When one of these buttons are selected, OUCIP enters the iterative loop for improving the compensation. First OUCIP checks the DC gains of all elements of the compensation matrix. If any are below the desired specifications, OUCIP allows them to vary, meaning that gradients of any unsatisfied

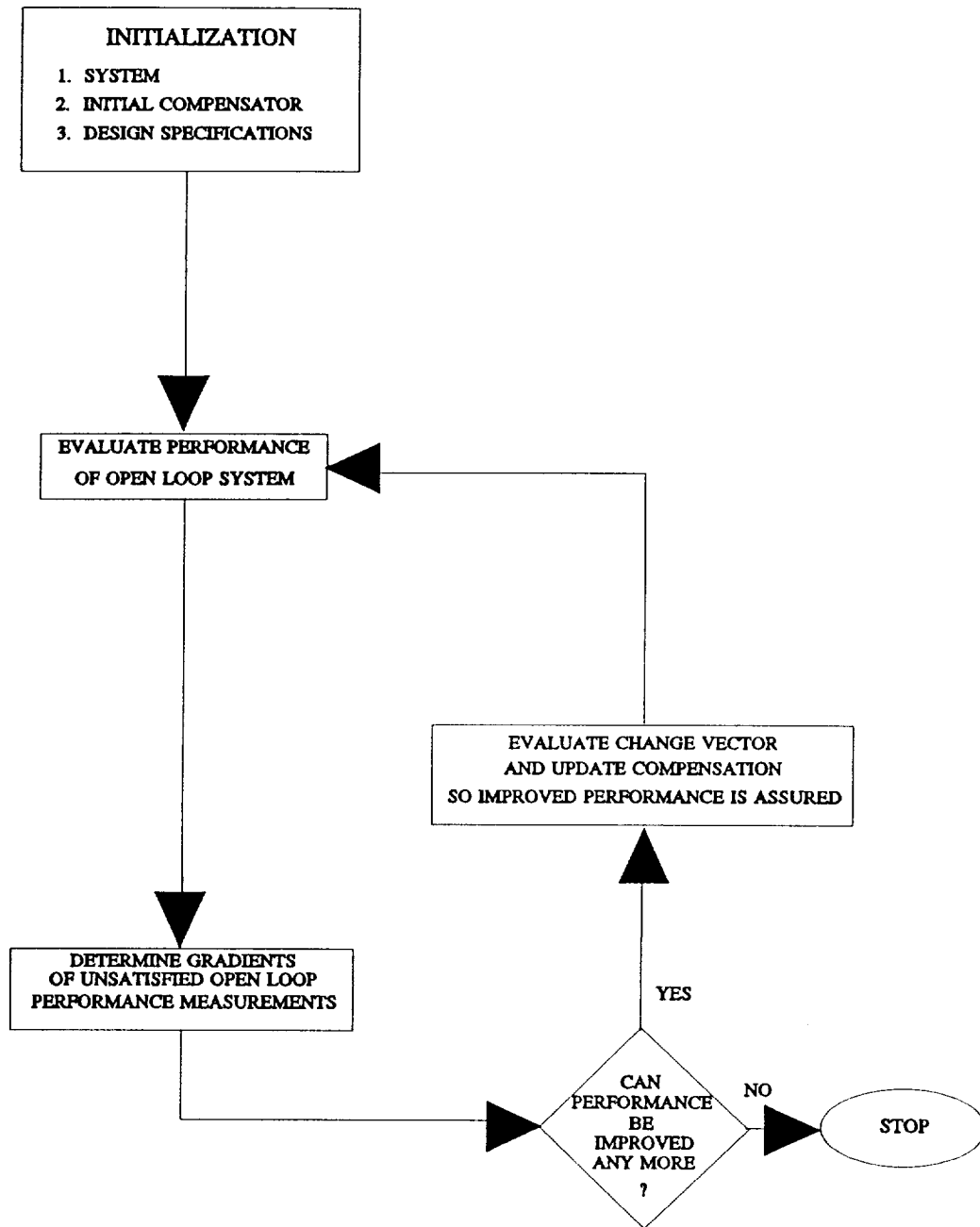


Figure 3.2: Simplified Flow Chart of original CIP

performance specification are computed and placed in respective positions of the gradient matrix. Next, OUCIP checks to see if the open loop specifications (Relative Stability) are "turned on" from the ACTIVATE menu. By default, the Relative Stability is the only type of specifications initially "turned on". If they are on, OUCIP evaluates the performance of the open loop systems. If they are not on OUCIP skips the previous step. OUCIP then checks to see if closed loop specifications are "turned on". If they are "on", the performance of the closed loop system is evaluated and gradients of unsatisfied performance measurements are calculated. If closed loop specifications are not "turned on", then the previous step is skipped. The next step is checking the stability of the closed loop system using the generalized Nyquist criterion. This is done regardless of the status of the closed loop specifications (on/off), since the frequency response of the closed loop system is always calculated.

After checking the stability of the closed loop system, OUCIP again checks the status of the open loop specifications (on/off). If they are "on", the gradients of the unsatisfied open loop performance measurements are calculated. OUCIP then checks the status of the compensator damping ratios. If they are "turned on", OUCIP calculates the gradients of all that are below the desired specification. If they are "turned off", this step is omitted. Next, the status of compensator DC gains are determined. If they are "turned on", the gradients of all that are below the desired DC gain level are calculated. If they are not "on", then no DC gain gradients are calculated. After all the gradients have been calculated, OUCIP determines whether the overall performance can be improved further. If not, OUCIP returns to the main menu to await further instructions. If further improvement is possible, the change vector is evaluated and the compensation matrix is updated in a way that improved performance is assured. OUCIP then determines if it has completed the final iteration from the EXECUTION menu. If it has, it once again returns to the main menu for further instructions. Otherwise, the iteration counter is incremented and the above series of events is repeated. The user may also pause the execution between iterations from the execution control window. This allows the user to return to the main menu prematurely in order to make an alteration in the specifications, step size, tolerance level, etc.

3.2 Conversion of Plane for Discrete-Time Design Improvement

OUCIP performs discrete-time (z-plane) design in the w-plane. This is transparent to the user. Before the execution loop of OUCIP is entered, the user specifies in which plane the data is given. OUCIP uses this information to determine whether to perform conversions of the data. If the plant data is in the w-plane then CIP converts the z-plane compensation data to the w-plane and when the execution loop is completed it converts the data back to the z-plane for output. Otherwise CIP also converts the plant data from the z-plane to the w-plane (if the plant and compensation data are both in the w-plane or s-plane, no conversion is performed). These conversions are accomplished by using a bilinear transformation. The particular bilinear transformation used in OUCIP is

$$z = \frac{1 + \frac{T}{2}w}{1 - \frac{T}{2}w} . \quad (3.2)$$

3.2.1 Conversion From z-plane to w-plane

If compensation is given in the z-plane, the element (i,j) of the compensator matrix is

$$G_{ij}(z) = (\text{gain}) \frac{\prod_{l=1}^{N1} (ZA_l + ZB_l z) \prod_{l=1}^{N2} (ZC_l + ZD_l z + ZE_l z^2)}{\prod_{l=1}^{M1} (PA_l + PB_l z) \prod_{l=1}^{M2} (PC_l + PD_l z + PE_l z^2)} . \quad (3.3)$$

where $N1$, $N2$, $M1$, and $M2$ are the numbers of first and second order zeros and poles respectively in the given compensator matrix element.

Each element of the compensator matrix has two types of factors, first order and second order. A general first-order z-plane factor is

$$H(z) = a + bz . \quad (3.4)$$

Using the bilinear transformation above, the w-plane equivalent is

$$H(w) = \frac{\alpha + \beta \frac{T}{2} w}{1 - \frac{T}{2} w} , \quad (3.5)$$

where

$$\alpha = a + b \quad (3.6)$$

and

$$\beta = b - a . \quad (3.7)$$

The conversion of (3.3) is simplified if the given compensator matrix element has the same number of first order numerator and denominator factors i.e., $N1 = M1$. If this is the case, only the numerator of (3.5) needs to be calculated for each factor. To obtain the w-plane equivalent, all z-plane first order factors are replaced with the equivalent w-plane factors computed from the numerator of (3.5). If $N1$ and $M1$ are not equal, then based on the smaller of the two, "neutral" first order numerator or denominator factors are added to the w-plane data until the number of first order factors is the same. The "neutral" first order equivalent for the z to w-plane conversion is

$$0z + 1 \Leftrightarrow -\frac{T}{2}w + 1 , \quad (3.8)$$

where T is the sampling time. Then the w-plane equivalents of $N1$ and $M1$ are set equal to the larger of the two from the z-plane.

A general second-order z-plane factor is

$$H(z) = c + dz + ez^2 . \quad (3.9)$$

The w-plane equivalent is

$$H(w) = \frac{\gamma + \delta Tw + \epsilon \frac{T^2}{4} w^2}{\left[1 - \frac{T}{2}w\right]^2} , \quad (3.10)$$

where

$$\gamma = c + d + e , \quad (3.11)$$

$$\delta = e - c , \quad (3.12)$$

and

$$\epsilon = c - d + e . \quad (3.13)$$

Similar to the conversion of the first-order factors, it can be seen that if the number of second-order numerator and denominator factors are the same, then the denominator from (3.10) can be neglected. In this case each z-plane second-order factor is replaced with the numerator equivalent from (3.10). If $N2$ and $M2$ are not equal, two first order "neutral" numerator or denominator factors are added in the respective place until the numbers are the same. The number of "neutral" first order factors introduced are added to $N1$ if the factors are numerator factors or $M1$ if they are denominator factors. Now that the conversion to the w-plane is completed, the data may be used by OUCIP along with w-plane plant frequency response data to improve the design.

3.2.2 Conversion from w-plane to z-plane

A general CIP compensator matrix element (ij) in the w-plane is

$$G_{ij}(w) = (\text{gain}) \frac{\prod_{l=1}^{N1} (ZA_l + ZB_l w) \prod_{l=1}^{N2} (ZC_l + ZD_l w + ZE_l w^2)}{\prod_{l=1}^{M1} (PA_l + PB_l w) \prod_{l=1}^{M2} (PC_l + PD_l w + PE_l w^2)} . \quad (3.14)$$

The data from the w-plane which is used in the design improvement process can be converted back to the z-plane each time output is generated if so desired. This is done by solving the bilinear transformation of (3.2) for w which gives

$$w = \frac{2(z-1)}{T(z+1)} . \quad (3.15)$$

This is the mapping used to convert data from the w-plane to the z-plane.

From inspection of (3.15), there are two types of w-plane factors, first and second order. A general w-plane first-order factor is

$$H(w) = \alpha + \beta w . \quad (3.16)$$

Using the bilinear transformation from (3.15), the z-plane equivalent of (3.16) is

$$H(z) = \frac{(a+bz)}{z+1} , \quad (3.17)$$

where

$$a = \alpha - \frac{2}{T}\beta , \quad (3.18)$$

and

$$b = \alpha + \frac{2}{T}\beta . \quad (3.19)$$

If the user initially starts the design improvement process with z-plane data, the order of the numerator and denominator of a given compensator element will be made equal in the conversion from the z-plane to the w-plane, and the denominator of (3.17) can be neglected. However if the user initially starts with w-plane data but wants the output to be given in the z-plane, addition of "neutral" factors from w to z-plane must be added. The first order "neutral" factor used to convert data from w to z-plane is

$$0w + 1 \Leftrightarrow z + 1 . \quad (3.20)$$

These "neutral" factors are used in a similar fashion as in the conversion from z to w -plane. They are added until the order of the numerator of a compensation element is the same as the order of the denominator of the same compensation element.

Thus, the conversion for first-order factors from the w -plane to the z -plane is the numerator of (3.17).

A general w -plane second order factor is

$$H(w) = \gamma + \delta w + \epsilon w^2 . \quad (3.21)$$

Using the bilinear transformation from (3.15), the z -plane equivalent is

$$H(z) = \frac{c + dz + ez^2}{(z + 1)^2} , \quad (3.22)$$

where

$$c = \gamma - \frac{2}{T} \delta + \frac{4}{T^2} \epsilon , \quad (3.23)$$

$$d = 2\gamma - \frac{8}{T^2} \epsilon , \quad (3.24)$$

and

$$e = \gamma + \frac{2}{T} \delta + \frac{4}{T^2} \epsilon . \quad (3.25)$$

Since the orders of the numerator and denominator of a compensator element were made to be equal, the denominator of (3.22) can be neglected. Thus, the conversion for second-order factors from the w -plane to the z -plane is the numerator of (3.22).

Using the bilinear transformation of (3.2) is used on (3.17) and (3.22), the true result is

$$H(w) = \frac{2\alpha + 2\beta w}{2} , \quad (3.26)$$

and

$$H(w) = \frac{4\gamma + 4\delta w + 4\epsilon w^2}{4} , \quad (3.27)$$

respectively.

Similarly, when the bilinear transformation of (3.15) is used in (3.5) and (3.10) the result is

$$H(z) = \frac{2a + 2bz}{2} , \quad (3.28)$$

and

$$H(z) = \frac{4c + 4dz + 4ez^2}{4} , \quad (3.29)$$

respectively.

Thus, since the orders of the numerator and denominator of a compensator element are equal, the twos in the denominators of the first order factors and the fours in the second order factors will cancel and the end result will be larger than the original data. Therefore, to have the end result correspond to the original data, each first order factor must be divided by two and each second order factor must be divided by four. This will ensure that the output data corresponds to the input data.

3.3 Improvement of Damping Ratios (ζ)

One of the goals of this work is to include the damping ratios (s-plane zetas) of second order factors of the compensator elements in both continuous and discrete or sampled-data systems as design specifications. Improving compensator damping ratios prevents excessive peaking or notching in the compensator frequency response and causes the closed-loop system to be more robust (less susceptible to data model errors). Lightly damped characteristics are not robust by nature because slight differences between the actual plant and the model used for design can cause serious performance and stability degradation of the closed loop system. The implications are that the ζ 's need to be treated as design specifications by CIP. In order for CIP to improve the ζ 's that are below design specifications, partial derivatives of these zetas must be computed and included in the matrix of gradient vectors, from which an appropriate directional vector is computed by CIP. As was mentioned earlier, the elements of the compensator matrix are assumed to be configured into ratios of cascade first and second order polynomials. For example the element (ij) has the s-plane general form

$$G_{ij}(s) = (\text{gain}) \frac{\prod_{l=1}^{N1} (ZA_l + ZB_l s) \prod_{l=1}^{N2} (ZC_l + ZD_l s + ZE_l s^2)}{\prod_{l=1}^{M1} (PA_l + PB_l s) \prod_{l=1}^{M2} (PC_l + PD_l s + PE_l s^2)}, \quad (3.30)$$

where the coefficients of the polynomials are assumed to be real.

The roots of a second order polynomial can be real or complex. The s-plane locations of the complex roots can be described using a damping ratio and a natural frequency. A damping ratio (ζ) is the cosine of the angle between the negative real axis and the line from the origin to a complex root in the s-plane and the natural frequency (ω_n), also known as the undamped frequency of oscillation, is the distance from the origin of the s-plane to the root. Therefore, only the second order terms of the elements of the compensator matrix are of concern in this chapter.

3.3.1 Calculation of Damping Ratios

A general form for an s-plane second order polynomial for the purposes of this discussion is

$$P(s) = a_2 s^2 + a_1 s + a_0. \quad (3.31)$$

Another form of (3.31) is

$$P(s) = a_2 \left[s^2 + 2\zeta\omega_n s + \omega_n^2 \right], \quad (3.32)$$

where ζ and ω_n are as earlier defined. Equating coefficients of (3.31) and (3.32) shows that

$$a_2 = a_2, \quad (3.33)$$

$$a_1 = 2a_2\zeta\omega_n, \quad (3.34)$$

$$a_0 = a_2\omega_n^2. \quad (3.35)$$

Solving (3.35) for ω_n gives

$$\omega_n = \sqrt{\frac{a_0}{a_2}}, \quad (3.36)$$

and from (3.34) ζ is found to be

$$\zeta = \frac{a_1}{2a_2\omega_n}. \quad (3.37)$$

Equations (3.36) and (3.37) provide the relationships for CIP to compute ζ from the s-plane coefficients of the second order terms. However, if a digital controller is being designed, the design must be done in the z-plane or w-plane. In this case, calculations for ζ are more challenging.

As mentioned earlier, CIP performs z-plane designs by converting real frequency to w-plane frequency and converting compensator data to the w-plane equivalents. In this case each element of the compensator matrix will appear as follows:

$$G_{ij}(w) = (\text{gain}) \frac{\prod_{l=1}^{N1} (ZA_l + ZB_l w) \prod_{l=1}^{N2} (ZC_l + ZD_l w + ZE_l w^2)}{\prod_{l=1}^{M1} (PA_l + PB_l w) \prod_{l=1}^{M2} (PC_l + PD_l w + PE_l w^2)}. \quad (3.38)$$

In order to calculate real ζ 's from the coefficients of w-plane second order terms, the mapping of the roots of a second order s-plane polynomial to the z-plane is considered, viz.,

$$(s+a)^2 + b^2 \Leftrightarrow z^2 - 2ze^{-aT} \cos(bT) + e^{-2aT} \quad (3.39)$$

in which

$$a = \zeta\omega_n \quad (3.40)$$

and

$$b = \sqrt{1-\zeta^2} \omega_n. \quad (3.41)$$

Thus, the s-plane second order polynomial of the form

$$P(s) = a_2 \left[s^2 + 2\zeta \omega_n s + \omega_n^2 \right] , \quad (3.42)$$

has a corresponding z-plane polynomial of the form

$$P(z) = a_2 \left[z^2 - 2e^{-\zeta \omega_n T} \cos(\sqrt{1-\zeta^2} \omega_n T) z + e^{-2\zeta \omega_n T} \right] . \quad (3.43)$$

which can be written as

$$P(z) = b_2 z^2 + b_1 z + b_0 , \quad (3.44)$$

where

$$b_2 = a_2 , \quad (3.45)$$

$$b_1 = a_2 \left[-2e^{-\zeta \omega_n T} \cos(\sqrt{1-\zeta^2} \omega_n T) \right] , \quad (3.46)$$

and

$$b_0 = a_2 \left[e^{-2\zeta \omega_n T} \right] . \quad (3.47)$$

The bilinear transformation used to get from the z-plane to the w-plane in CIP is

$$z = \frac{1 + \frac{T}{2} w}{1 - \frac{T}{2} w} . \quad (3.48)$$

Substituting into (3.44) gives

$$P(w) = b_2 \left(\frac{1 + \frac{T}{2} w}{1 - \frac{T}{2} w} \right)^2 + b_1 \left(\frac{1 + \frac{T}{2} w}{1 - \frac{T}{2} w} \right) + b_0 . \quad (3.49)$$

Simplifying produces

$$P'(w) = (b_2 - b_1 + b_0) \frac{T^2}{4} w^2 + (b_2 - b_0) T w + (b_2 + b_1 + b_0) , \quad (3.50)$$

where

$$P'(w) = \left(1 - \frac{T}{2} w \right)^2 P(w) . \quad (3.51)$$

Equation (3.50) can be written as

$$P'(w) = c_2 w^2 + c_1 w + c_0, \quad (3.52)$$

then, using (3.45), (3.46), and (3.47) in (3.50), and equating coefficients in (3.52) to (3.50) gives

$$c_2 = a_2 \left[1 + 2e^{-\zeta\omega_n T} \cos(\sqrt{1-\zeta^2}\omega_n T) + e^{-2\zeta\omega_n T} \right] \frac{T^2}{4}, \quad (3.53)$$

$$c_1 = a_2 (1 - e^{-2\zeta\omega_n T}) T, \quad (3.54)$$

$$c_0 = a_2 \left[1 - 2e^{-\zeta\omega_n T} \cos(\sqrt{1-\zeta^2}\omega_n T) + e^{-2\zeta\omega_n T} \right]. \quad (3.55)$$

The calculation of s-plane ζ and ω_n from the w-plane, second order coefficients begins with the equation for c_1 from (3.54). The known values are a_2 , c_1 , and T so the equation can be manipulated to give

$$\frac{a_2 T - c_1}{a_2 T} = e^{-2\zeta\omega_n T}. \quad (3.56)$$

Next, the square root is performed on (3.56) to give

$$\sqrt{\frac{a_2 T - c_1}{a_2 T}} = e^{-\zeta\omega_n T}. \quad (3.57)$$

The values, $e^{-\zeta\omega_n T}$ and $e^{-2\zeta\omega_n T}$, are substituted into the function c_0 of (3.55) which produces

$$c_0 = a_2 \left[1 - 2 \sqrt{\frac{a_2 T - c_1}{a_2 T}} \cos(\sqrt{1-\zeta^2}\omega_n T) + \frac{a_2 T - c_1}{a_2 T} \right]. \quad (3.58)$$

Assuming c_0 , c_1 , a_2 and T are known, (3.58) can be manipulated to produce from which ω_n can be calculated by dividing by T . Plugging (3.59) into (3.54) gives

$$\omega_n T = \frac{\cos^{-1} \left[\frac{(c_0 - 2a_2)T + c_1}{-2a_2 \sqrt{T \left[T - \frac{c_1}{a_2} \right]}} \right]}{\sqrt{1 - \zeta^2}} \quad (3.59)$$

$$c_1 = a_2 T - a_2 T e^{-2\zeta \left(\frac{\cos^{-1}(\cdot)}{\sqrt{1 - \zeta^2}} \right)} \quad (3.60)$$

where

$$(\cdot) = \frac{(c_0 - 2a_2)T + c_1}{-2a_2 \sqrt{T \left[T - \frac{c_1}{a_2} \right]}} \quad (3.61)$$

From this ζ can be computed. First, the natural logarithm of (3.60) is taken, and the equation is manipulated, resulting in

$$\ln \left[\frac{a_2 T - c_1}{a_2 T} \right] = \frac{-2\zeta \cos^{-1}(\cdot)}{\sqrt{1 - \zeta^2}} \quad (3.62)$$

Solving for ζ gives

$$\zeta = \pm \sqrt{\frac{\left[\frac{\ln(\cdot)}{\cos^{-1}(\cdot)} \right]^2}{4 + \left[\frac{\ln(\cdot)}{\cos^{-1}(\cdot)} \right]^2}} \quad (3.63)$$

where

$$(*) = \frac{a_2 T - c_1}{a_2 T} . \quad (3.64)$$

Solving for ω_n from (3.59) results in

$$\omega_n = \frac{\cos^{-1} \left[\frac{(c_0 - 2a_2)T + c_1}{-2a_2 \sqrt{T \left[T - \frac{c_1}{a_2} \right]}} \right]}{\sqrt{1 - \zeta^2} T} . \quad (3.65)$$

Thus, if a digital controller design is being done, CIP uses (3.63), (3.65), c_0 , c_1 , and c_2 to compute ζ 's for the second order terms.

3.3.2 Compensator Damping Ratio Improvement in the s-plane

There are two approaches that could be applied for improvement of the damping ratio (ζ). One is the "straight-forward" approach, where the partial derivatives of ζ w.r.t. the coefficients of the second order terms are simply calculated from (3.37). The alternative method is a "back-door" approach, which is described in the following text. In the latter method, the partial derivatives of equations (3.33), (3.34), and (3.35) w.r.t. ζ and ω_n are computed and manipulated to calculate the partial derivatives of ζ w.r.t. the coefficients of the second order terms. The "back-door" approach simplifies the calculations of partial derivatives of ζ 's when a digital controller design is done.

The partial derivatives of a second order term's coefficients, given in (3.33), (3.34), and (3.35), w.r.t. ζ and ω_n are

$$\frac{\partial a_2}{\partial \zeta} = 0, \quad \frac{\partial a_1}{\partial \zeta} = 2a_2\omega_n, \quad \frac{\partial a_0}{\partial \zeta} = 0 \quad (3.66)$$

$$\frac{\partial a_2}{\partial \omega_n} = 0, \quad \frac{\partial a_1}{\partial \omega_n} = 2a_2\zeta, \quad \frac{\partial a_0}{\partial \omega_n} = 2a_2\omega_n \quad (3.67)$$

However, partial derivatives of ζ and ω_n w.r.t. each coefficient are the desired end-product. These partial derivatives can be obtained from the differentials of the

coefficients. The differential of a function $f(q_1, q_2, \dots, q_n)$ w.r.t. the independent variables q_1, q_2, \dots, q_n is defined to be (E. Kreyszig, 1988)

$$df = \frac{\partial f}{\partial q_1} dq_1 + \frac{\partial f}{\partial q_2} dq_2 + \dots + \frac{\partial f}{\partial q_n} dq_n. \quad (3.68)$$

Using this definition where a_2, a_1 , and a_0 are assumed to be the dependent variables and ζ and ω_n are assumed to be the independent variables and placing the result in a matrix representation gives

$$\begin{bmatrix} da_0 \\ da_1 \\ da_2 \end{bmatrix} = \begin{bmatrix} \frac{\partial a_0}{\partial \zeta} & \frac{\partial a_0}{\partial \omega_n} \\ \frac{\partial a_1}{\partial \zeta} & \frac{\partial a_1}{\partial \omega_n} \\ \frac{\partial a_2}{\partial \zeta} & \frac{\partial a_2}{\partial \omega_n} \end{bmatrix} \begin{bmatrix} d\zeta \\ d\omega_n \end{bmatrix}. \quad (3.69)$$

Defining

$$M = \begin{bmatrix} \frac{\partial a_0}{\partial \zeta} & \frac{\partial a_0}{\partial \omega_n} \\ \frac{\partial a_1}{\partial \zeta} & \frac{\partial a_1}{\partial \omega_n} \\ \frac{\partial a_2}{\partial \zeta} & \frac{\partial a_2}{\partial \omega_n} \end{bmatrix}, \quad (3.70)$$

forming the pseudoinverse of M , and solving for $d\zeta$ and $d\omega_n$ produces

$$\begin{bmatrix} d\zeta \\ d\omega_n \end{bmatrix} = (M^T M)^{-1} M^T \begin{bmatrix} da_0 \\ da_1 \\ da_2 \end{bmatrix}. \quad (3.71)$$

Thus, each element of the pseudoinverse of M is a partial derivative of ζ or ω_n w.r.t. a coefficient of the given polynomial. In essence, ζ and ω_n have become the dependent variables and the coefficients of the polynomial have become the independent variables. These partial derivatives are used to form the gradient vector

of ζ 's and ω_n 's, which are used by CIP, along with other gradient vectors, to compute the directional change vector.

3.3.3 ζ Improvement in the z-plane or w-plane

Keeping with the earlier mentioned "back-door" approach, the partial derivatives of a second order term's coefficients, given in (3.53), (3.54), and (3.55), w.r.t. ζ and ω_n are

$$\frac{\partial c_2}{\partial \zeta} = -\frac{a_2 \omega_n T^3}{2} e^{-\zeta \omega_n T} \left[\cos(\sqrt{1-\zeta^2} \omega_n T) - \frac{\zeta}{\sqrt{1-\zeta^2}} \sin(\sqrt{1-\zeta^2} \omega_n T) + e^{-\zeta \omega_n T} \right], \quad (3.72)$$

$$\frac{\partial c_1}{\partial \zeta} = 2a_2 \omega_n T^2 e^{-2\zeta \omega_n T}, \quad (3.73)$$

$$\frac{\partial c_0}{\partial \zeta} = 2a_2 \omega_n T e^{-\zeta \omega_n T} \left[\cos(\sqrt{1-\zeta^2} \omega_n T) - \frac{\zeta}{\sqrt{1-\zeta^2}} \sin(\sqrt{1-\zeta^2} \omega_n T) - e^{-\zeta \omega_n T} \right], \quad (3.74)$$

$$\frac{\partial c_2}{\partial \omega_n} = -\frac{a_2 T^3}{2} e^{-\zeta \omega_n T} \left[\zeta \cos(\sqrt{1-\zeta^2} \omega_n T) + \sqrt{1-\zeta^2} \sin(\sqrt{1-\zeta^2} \omega_n T) + \zeta e^{-\zeta \omega_n T} \right], \quad (3.75)$$

$$\frac{\partial c_1}{\partial \omega_n} = 2a_2 \zeta T^2 e^{-2\zeta \omega_n T}, \quad (3.76)$$

and

$$\frac{\partial c_0}{\partial \omega_n} = 2a_2 T e^{-\zeta \omega_n T} \left[\zeta \cos(\sqrt{1-\zeta^2} \omega_n T) + \sqrt{1-\zeta^2} \sin(\sqrt{1-\zeta^2} \omega_n T) - \zeta e^{-\zeta \omega_n T} \right]. \quad (3.77)$$

Equations (3.45) - (3.50) provide the needed information for forming the matrix defined by (3.70) (c 's replace the a 's in this case). Then the partial derivatives of ζ and ω_n w.r.t. the coefficients c_2 , c_1 , and c_0 can be calculated using (3.71). Partial derivatives w.r.t. coefficients that are not part of a particular second order term are defined to be zero. Hence, in forming a gradient vector w.r.t. a ζ or ω_n , the only non-zero partial values are those terms corresponding to the coefficients of the respective second order factor of the ζ and ω_n of interest.

3.3.4 Comparison of s-plane ζ 's and w-plane ζ 's

The calculations performed in the latter part of section 3.1 were done in order to relate s-plane damping ratios to w-plane coefficients. If the calculations for ζ , performed in the first part of section 3.1, were performed with w-plane compensators, the result would be ζ_w , which is different from ζ . As mentioned earlier, in the s-plane ζ is defined to be the cosine of the angle between the negative real axis and the line from the origin to a complex root. Therefore, any roots lying on the same ray from the origin in the s-plane will have the same damping ratio (ζ). These rays are called constant zeta contours.

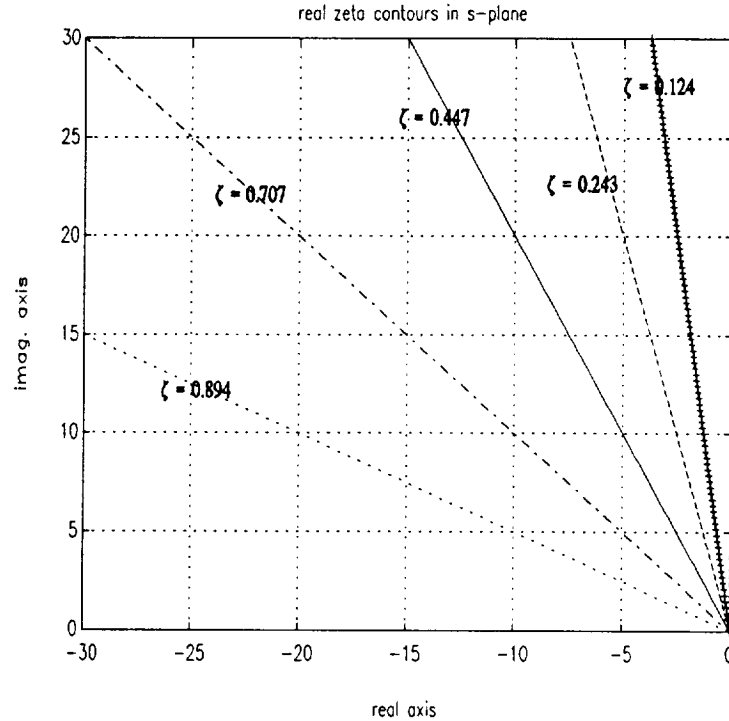


Figure 3.5: Family of constant zeta contours in the s-plane

Figure 3.5 shows several constant zeta contours and the respective zetas. Mapping the zeta lines shown in Figure 3.5 to the w-plane results in the curved contours shown in Figure 3.6.

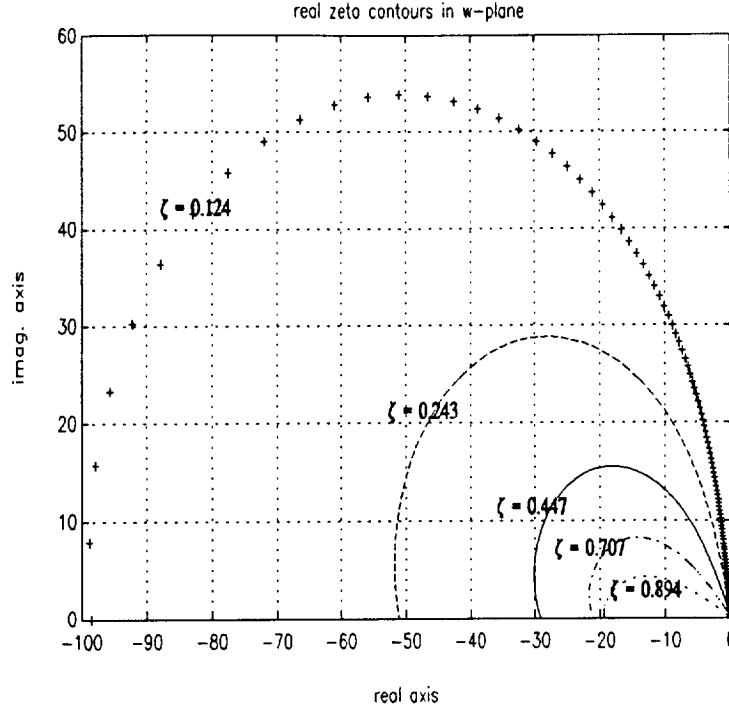


Figure 3.6: Family of constant zeta contours in the w-plane

The s-plane polynomial considered is

$$P(s) = s^2 + 2\zeta\omega_n s + \omega_n^2. \quad (3.78)$$

and the w-plane polynomial is

$$P(w) = w^2 + 2\zeta_w\omega_{w_n} w + \omega_{w_n}^2, \quad (3.79)$$

where ω_{w_n} is the w-plane natural frequency and ζ_w is the w-plane damping ratio. If Figure 3.5 was a family of contours in the w-plane, these lines would represent ζ_w . In Figure 3.7 three of the contours in Figure 3.5 were taken as ζ_w contours and the corresponding value of s-plane ζ contours were overlaid to show the differences between ζ and ζ_w in the w-plane.

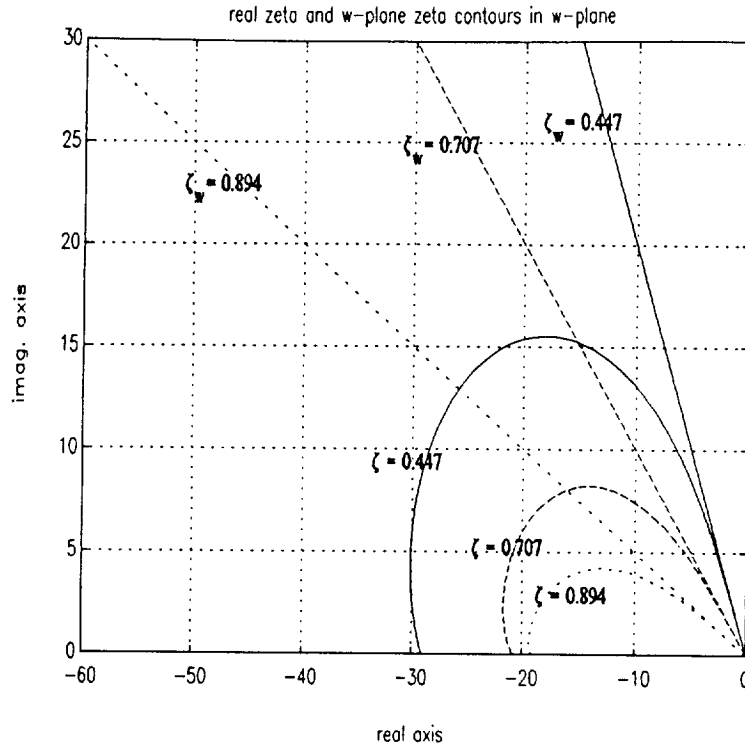


Figure 3.7: ζ_w contours and ζ contours of the same values in the w-plane

Notice that real ζ 's are restricted to a much smaller area of the complex plane than the ζ_w 's of the same value. The sampling period used in the above figures is .1 second, but if a different sampling period is chosen, the mappings change substantially. Figure 3.9 shows the effect of changing the sampling period while the ζ 's remain the same. Notice that as the sampling period decreases, the difference between the corresponding s-plane and w-plane contours becomes smaller.

If Figure 3.7 is mapped to the s-plane, it can be seen that the ζ 's are more restricted than the ζ_w 's here also. This is shown in Figure 3.8.

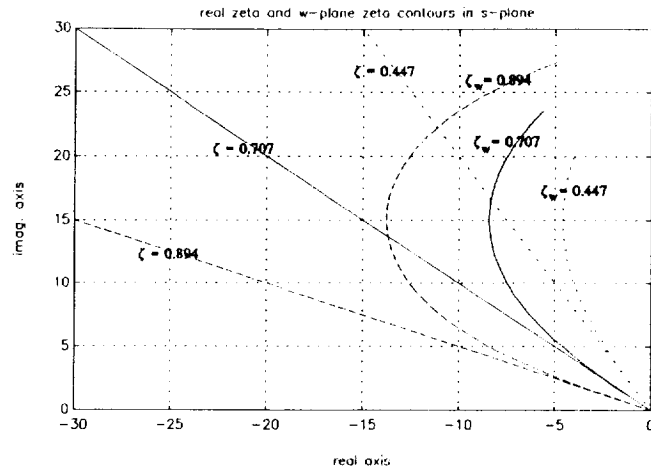


Figure 3.8: ζ and ζ_w contours in the s-plane.

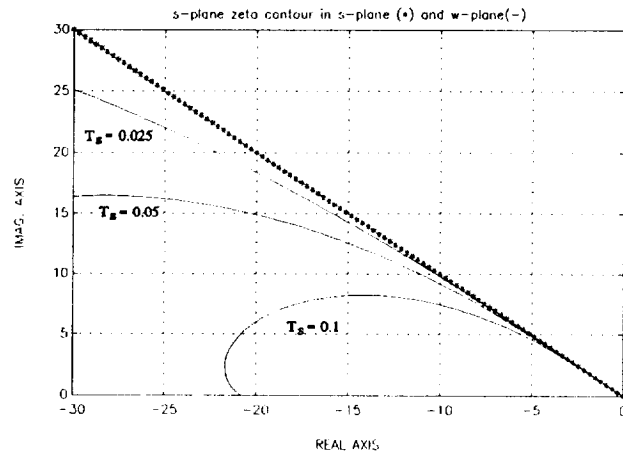


Figure 3.9: The effects of changing sampling period.

Thus it can be said that s-plane damping ratio constraints are more difficult to achieve in digital control system designs than w-plane zetas constraints.

3.4 Compensator DC Gain Improvement

DC gains are the gains of any given system at the point when frequency is equal to zero. Increasing the compensator DC gains and consequently the loop DC gains, allows for the improvement of steady-state error and disturbance rejection performance. As mentioned earlier, previous versions of CIP did not allow the user to set a desired value for the final DC gains. The user was given the option to let the DC gains vary with the improvements of the selected margins or to constrain the DC gains to remain constant. Thus, the best guaranteed values were the original gains. In order to improve selected DC gains to exceed a specified value, each gain must be treated as a margin and the gradient vector of the gains w.r.t. the compensator coefficients of the given compensator matrix element must be computed and included in the gradient matrix.

The DC gain of a system is calculated by setting the frequency ω (rad/sec) to zero and calculating the value of the transfer function. In the s-plane, where s is simply set to zero. In the w-plane, w is set to zero since ω_w is related to ω through a tangent function (weighted by a constant, depending on the bilinear transformation used). In the z-plane, $z=1$ since $z=e^{sT}$. However, since CIP performs digital controller designs in the w-plane, only the s-plane and w-plane cases are considered.

The general compensator matrix element (ij) in the s-plane is

$$G_{ij}(s) = (\text{gain}) \frac{\prod_{i=1}^{N1} (ZA_i + ZB_i s) \prod_{j=1}^{N2} (ZC_j + ZD_j s + ZE_j s^2)}{\prod_{i=1}^{M1} (PA_i + PB_i s) \prod_{j=1}^{M2} (PC_j + PD_j s + PE_j s^2)} . \quad (3.80)$$

Likewise, in the w-plane the general compensator matrix element (ij) is

$$G_{ij}(w) = (\text{gain}) \frac{\prod_{i=1}^{N1} (ZA_i + ZB_i w) \prod_{j=1}^{N2} (ZC_j + ZD_j w + ZE_j w^2)}{\prod_{i=1}^{M1} (PA_i + PB_i w) \prod_{j=1}^{M2} (PC_j + PD_j w + PE_j w^2)} . \quad (3.81)$$

From (3.80) and (3.81) it can be seen that if s or w are set to zero respectfully, the DC gains will become

$$G_{ij}(0) = (\text{gain}) \frac{\prod_{i=1}^{N1} ZA_i \prod_{j=1}^{N2} ZC_j}{\prod_{i=1}^{M1} PA_i \prod_{j=1}^{M2} PC_j} \quad (3.82)$$

in both cases, where it is understood that the coefficients of the two planes are different for equivalent compensation.

The values used to calculate the directional vector are also quite easy to calculate. They are the partial derivatives of (3.82) w.r.t. each of its coefficients. The partial derivative of the compensator element (ij) w.r.t. the kth ZA term is calculated by dividing the DC gain by the kth ZA term, thus

$$\frac{\partial G_{ij}(0)}{\partial ZA_k} = (gain) \frac{\prod_{i=1}^{N1} ZA_i \prod_{j=1}^{N2} ZC_j}{\prod_{i=1}^{M1} PA_i \prod_{j=1}^{M2} PC_j} \quad (3.83)$$

Similarly, the partial derivative of the DC gain with respect to the kth ZC term is

$$\frac{\partial G_{ij}(0)}{\partial ZC_k} = (gain) \frac{\prod_{l=1}^{N1} ZA_l \prod_{l=1}^{N2} ZC_l}{\prod_{l=1}^{M1} PA_l \prod_{l=1}^{M2} PC_l} \quad (3.84)$$

The partial derivative of the DC gain with respect to the kth PA denominator term is calculated by dividing the DC gain by the negative value of the kth PA term. This results in

$$\frac{\partial G_{ij}(0)}{\partial PA_k} = - \frac{(gain)}{PA_k} \frac{\prod_{l=1}^{N1} ZA_l \prod_{l=1}^{N2} ZC_l}{\prod_{l=1}^{M1} PA_l \prod_{l=1}^{M2} PC_l} \quad (3.85)$$

and similarly the partial derivative of the DC gain with respect to the kth PC term is

$$\frac{\partial G_{ij}(0)}{\partial PC_k} = - \frac{(gain)}{PC_k} \frac{\prod_{l=1}^{N1} ZA_l \prod_{l=1}^{N2} ZC_l}{\prod_{l=1}^{M1} PA_l \prod_{l=1}^{M2} PC_l} \quad (3.86)$$

These values are inserted into the corresponding locations in the partial derivative matrix and used by CIP to calculate the directional vector for improving the controller.

3.5 Analysis of Closed Loop Stability

In order to analyze stability of multivariable closed loop systems the generalized Nyquist stability criterion can be used (J.M. Maciejowski, 1989). Generalized Nyquist's Stability Criterion relates absolute stability of a closed loop linear system using the number of open loop poles in the RHP and the frequency response of the determinant of the return difference matrix $(I+GH(s))$. Closed loop stability can be determined graphically by examining the plot of this frequency response. For the closed loop system to be stable all poles of the closed loop system must lie in the left half plane. CIP assumes that the initial compensation produces a closed loop system that is stable.

Theoretically, because CIP iteratively improves system performance, closed loop stability should be maintained. However, because the process is numerical and the amount of data is finite, closed loop stability can be lost. In order to check for this and warn the user, CIP has been given the capability to check for closed loop stability. The implementation is described in this sub-section.

3.5.1 Nyquist's Mapping Theorem (Generalized for Multivariable Systems)

Let $F(s)$ be a matrix whose elements are transfer functions, with Z being the number of zeros of the $\det[F(s)]$ that lie inside some closed contour in the s -plane and P being the number of poles of the $\det[F(s)]$ that lie inside the same closed contour in the s -plane. Nyquist's mapping theorem states that by mapping this closed contour of the s -plane into the $\det[F(s)]$ -plane as a closed contour, the total number of clockwise encirclements, N , of the origin of the $\det[F(s)]$ -plane by this closed contour, is equal to the difference of the number of zeros and poles: $N = Z - P$ (K. Ogata, 1990).

This mapping theorem can be proven by the principle of the argument (R.V. Churchill, et.al., 1990). This theorem is useful because by plotting the frequency response of $\det[F(s)]$, N , can be counted and P can be found from the $F(s)$ matrix. Thus, Z can be found by $Z = N + P$ where Z is the number of zeros of the $\det[F(s)]$ in the right half s -plane. This mapping theorem can now be applied to control system stability analysis.

3.5.2 Using the Nyquist Mapping Theorem in Control System Stability Analysis

Consider the multivariable system in Figure 5.1, where

$$G_C(s) = \begin{bmatrix} g_{c_{11}} & g_{c_{12}} & \dots & g_{c_{1m}} \\ g_{c_{21}} & g_{c_{22}} & \dots & g_{c_{2m}} \\ \vdots & \vdots & \dots & \vdots \\ g_{c_{n1}} & g_{c_{n2}} & \dots & g_{c_{nm}} \end{bmatrix} \quad (3.87)$$

and

$$G_P(s) = \begin{bmatrix} g_{p_{11}} & g_{p_{12}} & \dots & g_{p_{1n}} \\ g_{p_{21}} & g_{p_{22}} & \dots & g_{p_{2n}} \\ \vdots & \vdots & \dots & \vdots \\ g_{p_{m1}} & g_{p_{m2}} & \dots & g_{p_{mn}} \end{bmatrix} . \quad (3.88)$$

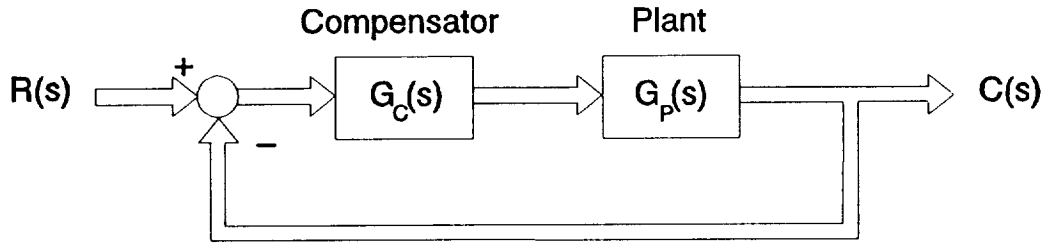


Figure 3.10: General Multi-input Multi-output (MIMO) System.

Then $G_P(s)G_C(s)$ is a square matrix of dimensions m by m referred to simply as $G(s)$. From Figure 3.10 it can be seen that

$$C(s) = G(s)[R(s) - C(s)] . \quad (3.89)$$

Thus,

$$C(s) = [I + G(s)]^{-1} G(s) R(s) .$$

From (3.90), the closed-loop transfer function matrix, $T(s)$, is

$$T(s) = [I + G(s)]^{-1} G(s) .$$

It is easily determined that $\det[I + G(s)]$ has the same poles as $\det[G(s)]$.

Using the Nyquist stability criterion described in section 5.1 on $\det[I + G(s)]$, assuming that the closed contour of the s -plane encircles the entire right half plane (RHP), (see Figure 3.11) where P is the number of RHP poles of $\det[I + G(s)]$. By examining the plot of the frequency response of $\det[I + G(s)]$ and counting the encirclements of the origin N can be determined. Then applying the Nyquist stability criterion $Z = N + P$, the number of zeros of $\det[I + G(s)]$ can be found. This is the number of RHP poles of the closed loop system, and for stability must be zero. For example, if there are two RHP open loop poles, there must be two counter-clockwise encirclement of the origin of the $\det[I + G(s)]$ plane to ensure closed loop stability (shown in Figure 3.11 and Figure 3.12).

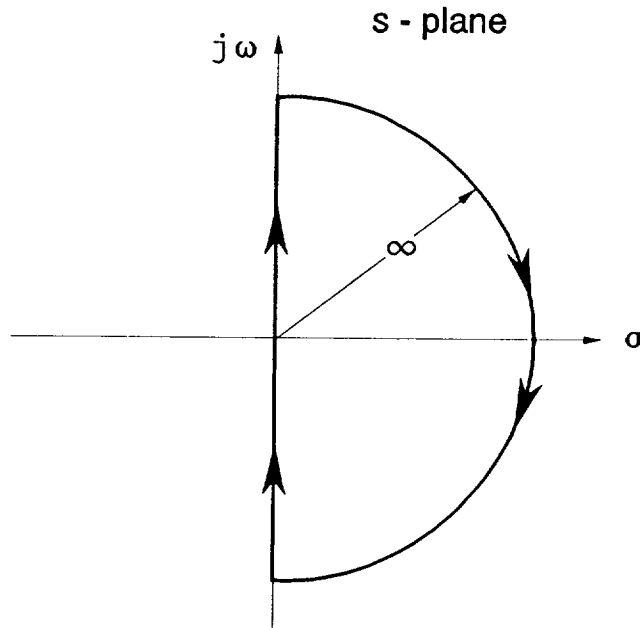


Figure 3.11: Closed contour of s -plane encircling the RHP

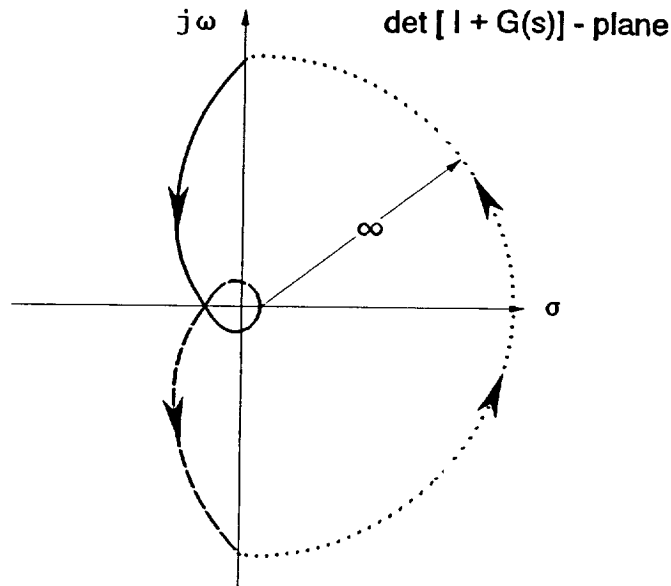


Figure 3.12: Nyquist Plot of $\det[I + G(s)]$ with 2 counter-clockwise encirclements of the origin

3.5.3 Discussion of CIP's Implementation of the Nyquist Stability Criterion

CIP uses frequency response data that is often obtained by experimentation. Therefore there may be no knowledge of the number of open loop RHP poles. CIP assumes that the initial compensator produces a closed loop system to be stable. Using this assumption and the data of the frequency response of $\det[I + G(s)]$ on the zeroth iteration of CIP, the number of open loop RHP poles are initialized to $-N$ (the number of encirclements of the origin of the $\det[I + G(s)]$). This does not represent the true number of open loop RHP poles since the frequency response of $\det[I + G(s)]$ is not mirrored about the real axis (i.e., CIP does not form the negative frequency portion of the polar frequency response), and no infinite semicircles are added (representing the number of poles on the Nyquist Contour). Although CIP uses the Nyquist stability criterion, it does not use a complete Nyquist Diagram.

CIP determines the number of encirclements of the origin of $\det[I + G(s)]$, or encirclements of $-1 + j0$ point of $\det[I + G(s)] - 1$ (this is the data OUCIP actually plots as graphical output because the origin of a plot with magnitudes in dB is difficult to determine), by counting crossings of the individual axes. If the positive, real axis is crossed once from above and then once from below, the net crossings are zero. Likewise each of the other axes are checked similarly. If any axis has a net crossing that is not equal to zero, then all the net crossings are compared and the maximum (or minimum if crossings are counter clockwise) value is taken to be the number of encirclements. This is shown in Figure 3.12. In future iterations the number of

encirclements, N , is added to the number P found in the initial iteration. If the resulting number, Z , is positive, CIP assumes the closed loop system has become unstable. If the resulting number, Z , is negative, the number of open loop RHP poles is re-initialized to $-N$ (the new number of encirclements).

Note: CIP uses a finite number of data points. It is assumed that the user has accounted for this by giving CIP sufficient data. Otherwise the frequency response of $\det[I + G(s)]$ may be inaccurate and crossings may be counted that should not be. This will cause CIP to determine that the closed loop system has (a) become unstable or (b) that the number of open loop RHP poles must be re-initialized (depending on direction of encirclements). Either of which will cause CIP to warn the user of an instability that has not actually occurred!

3.6 Improvement of closed loop disturbance rejection characteristics

Disturbance rejection specifications have also been added to CIP. It is now possible to specify a maximum allowable RMS and/or peak values for any element of the close loop frequency response matrix from disturbance inputs D to measured outputs Y . The theory necessary for the implementation of these specifications in CIP follows.

With the system set up as in Figure 3.1, the transfer function matrix from D to Y is

$$G_{Y/D} = [I + G_{22}G_c]^{-1} G_{21} \quad , \quad (3.92)$$

where the dependence on the transform variable has been dropped for convenience.

Let the kl th element of G_c be a function of the parameter γ , in this case a compensator coefficient. Then

$$\begin{aligned} \frac{\partial G_{Y/D}}{\partial \gamma} &= \frac{\partial [I + G_{22}G_c]^{-1}}{\partial \gamma} G_{21} \\ &= -[I + G_{22}G_c]^{-1} G_{22} e_k e_l^T [I + G_{22}G_c]^{-1} G_{21} \frac{\partial G_{c_{kl}}}{\partial \gamma} \quad , \end{aligned} \quad (3.93)$$

where e_i is an elementary column vector of appropriate dimension that has 1 as its i th element and zeros in all other elements, and $G_{c_{kl}}$ is the kl th element of G_c . If the ij th element of $G_{Y/D}$ is denoted as $[G_{Y/D}]_{ij}$, then

$$\frac{\partial [G_{Y/D}]_{ij}}{\partial \gamma} = e_i^T \frac{\partial G_{Y/D}}{\partial \gamma} e_j \quad . \quad (3.94)$$

An expression for the partial derivative of the magnitude of the ij th element of $G_{Y/D}$ with respect to a parameter γ of the compensator is then given by

$$\frac{\partial |G_{Y/D}|_{ij}}{\partial \gamma} = \frac{\operatorname{Re} \left\{ \left([G_{Y/D}]_{ij} \right) \left[\frac{\partial [G_{Y/D}]_{ij}}{\partial \gamma} \right]^* \right\}}{\sqrt{([G_{Y/D}]_{ij})([G_{Y/D}]_{ij}^*)}} . \quad (3.95)$$

For each peak detected in each element of the frequency response matrix $G_{Y/D}$ that exceeds the maximum allowed and for each compensator parameter, equation (3.95) is evaluated at the corresponding frequency and placed in the gradient vector of the violated peak value.

The evaluation of gradient vectors for violated RMS values relies on the following derivation. Let

$$h_k = [G_{Y/D}(e^{j\omega_k T})]_{ij} , \quad k=1,2,\dots,N , \quad (3.96)$$

where ω denotes frequency in rad/sec and N is the number of frequency points available. Using a trapezoidal integrator to compute the RMS value, rms_{ij} , of the ij th element of the frequency response matrix from D to Y yields

$$rms_{ij} = \sqrt{S_1/2\pi} , \quad (3.97)$$

where

$$S_1 = |h_1|^2(\omega_2 - \omega_1) + \sum_{l=2}^{N-1} |h_l|^2(\omega_{l+1} - \omega_{l-1}) + |h_N|^2(\omega_N - \omega_{N-1}) . \quad (3.98)$$

Letting

$$S_2 = h_1 \frac{\partial h_1^*}{\partial \gamma} (\omega_2 - \omega_1) + \sum_{l=2}^{N-1} h_l \frac{\partial h_l^*}{\partial \gamma} (\omega_{l+1} - \omega_{l-1}) + h_N \frac{\partial h_N^*}{\partial \gamma} (\omega_N - \omega_{N-1}) , \quad (3.99)$$

then the partial derivative of rms_{ij} with respect to the compensator parameter γ is

$$\frac{\partial rms_{ij}}{\partial \gamma} = \frac{\operatorname{Re}(S_2)}{2\pi rms_{ij}} . \quad (3.100)$$

For each RMS violation in the closed loop transfer function matrix from D to Y , equation (3.100) is evaluated and the result is placed in the corresponding gradient vector.

3.7 Examples and Results

In this section, the capabilities of CIP are demonstrated using examples of design improvements in all three planes; z-plane, s-plane and w-plane. The examples performed in the s-plane and z-plane are based on a simple two-axis pointing system shown in Figure 3.13. The example performed in the w-plane uses NASA supplied data for the Space Shuttle.

In order to achieve the desired improvements, the Ohio University version of CIP is used. This version of CIP, known as OUCIP, represents a major overhaul of CIP that includes the work in this thesis plus enhancements recently developed by others that include: (1) a new menu-driven, friendly front-end, (2) closed loop design improvements and (3) an improved directional vector routine. The user has the ability to stop the program at any iteration and alter OUCIP's direction of execution. There are many ways in which a user may change the course of the design. For example, OUCIP has an ACTIVATE menu. In this menu, the user can specify what types of design requirements to improve. There are currently four choices: (1) Relative Stability, which include gain margins, phase margins, attenuation levels and the stability margins (minimum loop-at-a-time return difference values), (2) Disturbance Rejection, which allows the user to set a maximum magnitude for the disturbance to output frequency response and RMS values, (3) Compensator DC Gain which allows the user to specify minimum DC gains of the compensator elements, and (4) Compensator Damping Ratios, where the user can define minimum damping ratios for the compensator elements.

3.7.1 Example 1: A Pointing System

Figure 3.13 shows a simple two-axis pointing system. This plant has torque control inputs T_{EL} and T_{AZ} , angular displacement outputs θ_{EL} and θ_{AZ} , and torque disturbance inputs D_{EL} and D_{AZ} . The shafts are assumed to be flexible and this is represented by lightly damped modes at 0.16 and 0.38 Hz, and rigid body modes. There is also significant coupling between input/output pairs. The plant is described by 200 points for each of the 4 open-loop frequency responses. The interaction between disturbance inputs and measured outputs are also described by 200 points for each of the 4 open-loop frequency responses. For the z-plane case, a sampled data system with a sampling time of 0.1 seconds is used. In the s-plane case, a theoretical continuous time system is used.

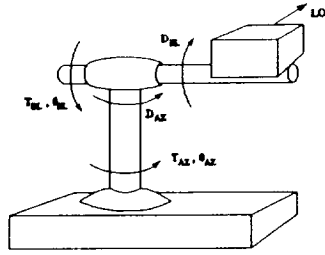


Figure 3.13 Two-Axis Pointing System

For each case, an initial diagonal compensator was designed by using two cascade first order lead stages in element 1,1 and a cascade combination of a first order and second order lead stages in element 2,2 in order for the system to resemble a co-located sensor-actuator pair. While the initial compensator did not meet all design specifications, it did stabilize the closed loop system. CIP was selected to modify the compensator to meet the desired design requirements. The following subsections show the designs performed in both the s-plane and z-plane.

3.7.2 Pointing System Example (s-plane)

Figure 3.14 shows a block diagram of the pointing system used in this s-plane example. Note that closed loop disturbance rejection was not used as a design specification for this example.

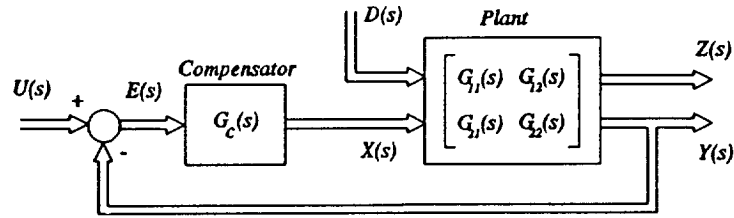


Figure 3.14: Pointing System Block Diagram

Figures 3.15 to 3.22 show the magnitude and phase frequency response plots of the uncompensated open loop plant described by the 200 frequency points mentioned previously.

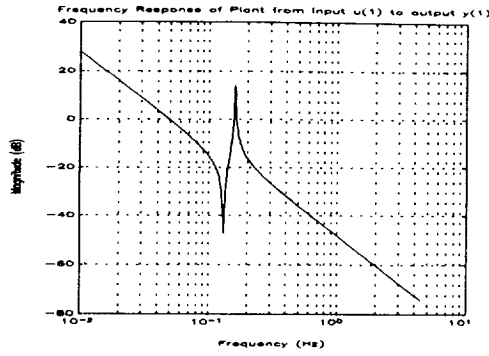


Figure 3.15: Magnitude Frequency Response of Plant from $x(1)$ to $y(1)$

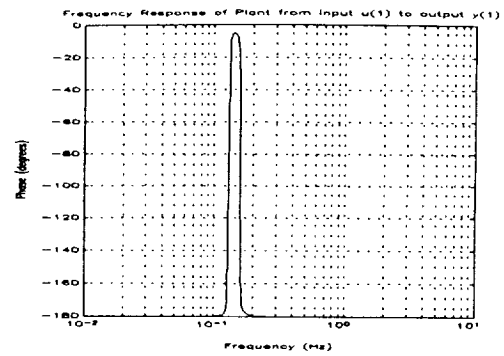


Figure 3.16: Phase Frequency Response of Plant from $x(1)$ to $y(1)$

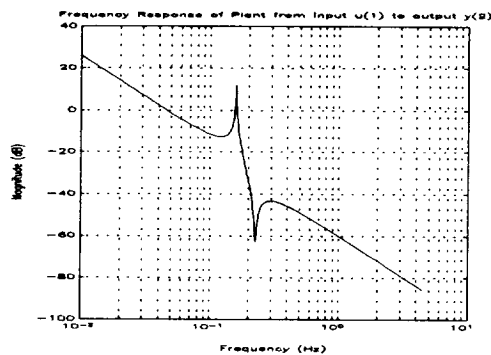


Figure 3.17: Magnitude Frequency Response of Plant from $x(1)$ to $y(2)$

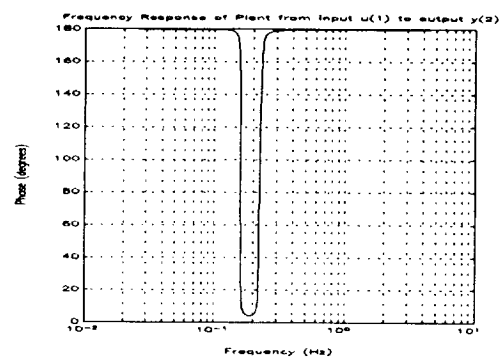


Figure 3.18: Phase Frequency Response of Plant from $x(1)$ to $y(2)$

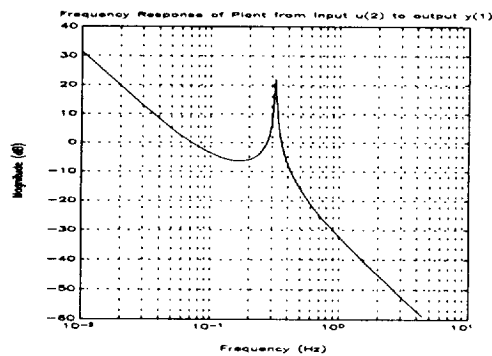


Figure 3.19: Magnitude Frequency Response of Plant from $x(2)$ to $y(1)$

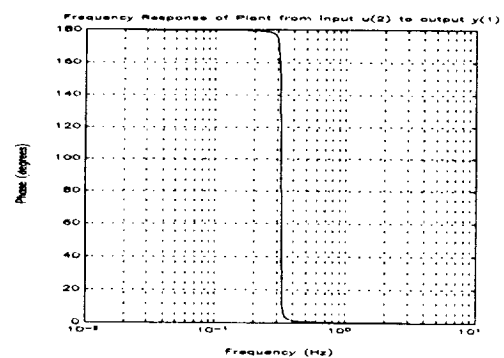


Figure 3.20: Phase Frequency Response of Plant from $x(2)$ to $y(1)$

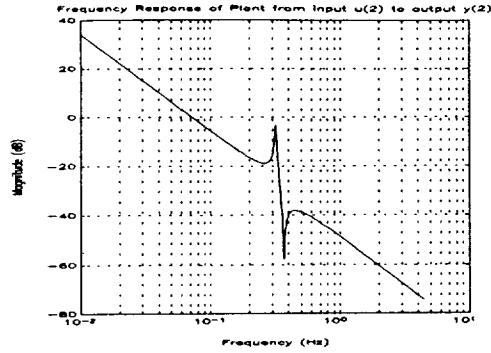


Figure 3.21: Magnitude Frequency Response of Plant from $x(2)$ to $y(2)$

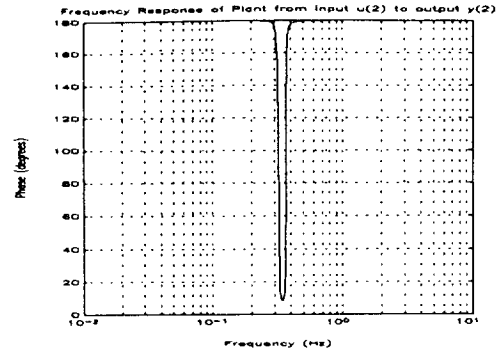


Figure 3.22: Phase Frequency Response of Plant from $x(2)$ to $y(2)$

For this example, the design specifications initially activated were relative stability, compensator DC gains, and compensator damping ratios. At the start of the design improvement, only the DC gains, damping ratios and one attenuation level were violated.

Table 3.1 and Table 3.2 show the initial and final compensator coefficients. Notice that the zeroth order coefficients of the numerators have increased in size while that of the denominator have decreased. These changes worked to increase the DC gains of the compensators. From Table 3.3 and Table 3.4, it seems that the compensator damping ratios and DC gains were the only improved specifications from iteration 0 to iteration 108, but in reality, CIP lets performance measurements that exceed specifications degrade as initially violated specifications are improved. Thus, many of the initially satisfied specifications become violated and are then improved. This allows CIP to use a "ratchet" effect. As violated specifications are improved, causing conflicting specifications to become violated, the directional vector is computed so as to update the compensation such that favorable changes result for all violated specifications. For example, in the s -plane, DC gains and damping ratios are conflicting constraints. Looking at (3.32), if the zeroth order coefficient is increased to improve DC gains, as is the case in a numerator second order polynomial, then the first order coefficient, still at the same value, has a lower contribution from the damping ratio. (This may be the reason that the denominator damping ratios generally have a much larger value than that of the numerator as can be seen from Table 3.1 and Table 3.2)

Table 3.1 Initial Compensator Elements, s-plane pointing example.

	Zeroth Order	First Order	Second Order
Numerator 1 Comp.(1,1)	1.00000	114.587	N/A
Denominator 1 Comp.(1,1)	1.00000	0.008727	N/A
Numerator 2 Comp.(1,1)	1.00000	0.470000	N/A
Denominator 2 Comp.(1,1)	1.00000	0.230000	N/A
Numerator 1 Comp.(2,2)	1.00000	114.587	N/A
Denominator 1 Comp.(2,2)	1.00000	0.008727	N/A
Numerator 2 Comp.(2,2)	1.00000	0.200000	1.00000
Denominator 2 Comp.(2,2)	1.00000	0.008300	0.173600

Table 3.2 Final (iteration 108) Compensator Elements, s-plane Pointing Example

	Zeroth Order	First Order	Second Order
Numerator 1 Comp.(1,1)	1.35056	114.581	N/A
Denominator 1 Comp.(1,1)	0.451931	0.1000×10^{-9}	N/A
Numerator 2 Comp.(1,1)	1.35090	0.756108	N/A
Denominator 2 Comp.(1,1)	0.451931	0.751140	N/A
Numerator 1 Comp.(2,2)	1.42224	114.585	N/A
Denominator 1 Comp.(2,2)	0.537412	0.0029069	N/A
Numerator 2 Comp.(2,2)	1.35587	0.677156	0.884631
Denominator 2 Comp.(2,2)	0.100799	0.559672	0.351598

Table 3.3 Initial Objective Function Values for the s-plane Pointing Example

Iteration	Type	Frequency (Hz)	Desired	Current
0	G1	0.2799	≥ 0.5000	0.7844
0	P1	0.06362	$\geq 50.0^\circ$	140.2°
0	P1	0.1337	$\geq 50.0^\circ$	125.2°
0	P1	0.2437	$\geq 50.0^\circ$	100.7°
0	P1	0.3041	$\geq 50.0^\circ$	68.69°
0	P1	0.6582	$\geq 50.0^\circ$	77.15°
0	A1	2.337	≤ 0.2000	0.1758
0	A1	4.405	≤ 0.2000	0.08588
0	P2	0.04497	$\geq 50.0^\circ$	125.5°
0	P2	0.1270	$\geq 50.0^\circ$	165.5°
0	P2	0.1444	$\geq 50.0^\circ$	167.7°
0	P2	0.2264	$\geq 50.0^\circ$	108.4°
0	P2	0.8480	$\geq 50.0^\circ$	64.70°
0	A2	2.337	≤ 0.2000	0.2304
0	A2	4.405	≤ 0.2000	0.1104
0	Z22 ζ	0.1592	≥ 0.3000	0.1000
0	P22 ζ	0.3820	≥ 0.3000	0.0100
0	DC11	0.0000	≥ 0.6000	0.07080
0	DC22	0.0000	≥ 0.6000	0.03162

Types: G_x, P_x, A_x: Gain Margin, Phase Margin, and Attenuation Level, respectively, in loop x. DC_{xy}, Z_{xy} ζ , P_{xy} ζ : DC gain, damping ratio of zero, damping ratio of pole, respectively, for compensator xy. DP_{xy}, DR_{xy}: peak and RMS value, respectively, for xy element of frequency response from D to Y.

Table 3.4 Final Objective Function Values for the s-plane Pointing Example

Iteration	Type	Frequency (Hz)	Desired	Current
108	G1	0.3655	≥ 0.5000	5.549
108	P1	0.06004	$\geq 50.0^\circ$	80.58°
108	P1	0.07142	$\geq 50.0^\circ$	128.9°
108	P1	0.7238	$\geq 50.0^\circ$	50.50°
108	A1	2.337	≤ 0.2000	0.1989
108	A1	4.405	≤ 0.2000	0.09871
108	P2	0.2437	$\geq 50.0^\circ$	89.83°
108	P2	0.2773	$\geq 50.0^\circ$	169.3°
108	P2	0.6582	$\geq 50.0^\circ$	59.97°
108	A2	2.337	≤ 0.2000	0.1816
108	A2	4.405	≤ 0.2000	0.09225
108	Z22 ζ	0.1970	≥ 0.3000	0.3091
108	P22 ζ	0.08522	≥ 0.3000	1.486
108	DC11	0.0000	≥ 0.6000	0.6324
108	DC22	0.0000	≥ 0.6000	1.126

Figure 3.23 and Figure 3.24 show the differences between the original and final frequency responses of the compensated open loop systems. Notice that the compensated open loop system 1 has lost loop integrity, in other words, the compensated open loop system has gone unstable, but by looking at Figure 3.25, the determinant of the return difference shows the same number of encirclements of the 0 dB at -180° (or $-1 + j0$) point as the original system, thus the closed loop system remains stable (assuming the initial design was stable).

The most distinctive changes can be seen by examining the frequency responses of the compensators themselves. The compensator element from error signal $e(1)$ to compensator output $x(1)$ shows a large increase in the lower frequency gain. This is due to the improvement of the DC gains. Figure 3.26 and Figure 3.27 show this improvement. The compensator element from error signal $e(2)$ to compensator output $x(2)$ initially has a very lightly damped pole/zero pair. The final compensator

frequency response is much smoother due to improved damping ratios. The low frequency gain has again been increased due to the improved DC gains. These changes are evident in Figure 3.28 and Figure 3.29.

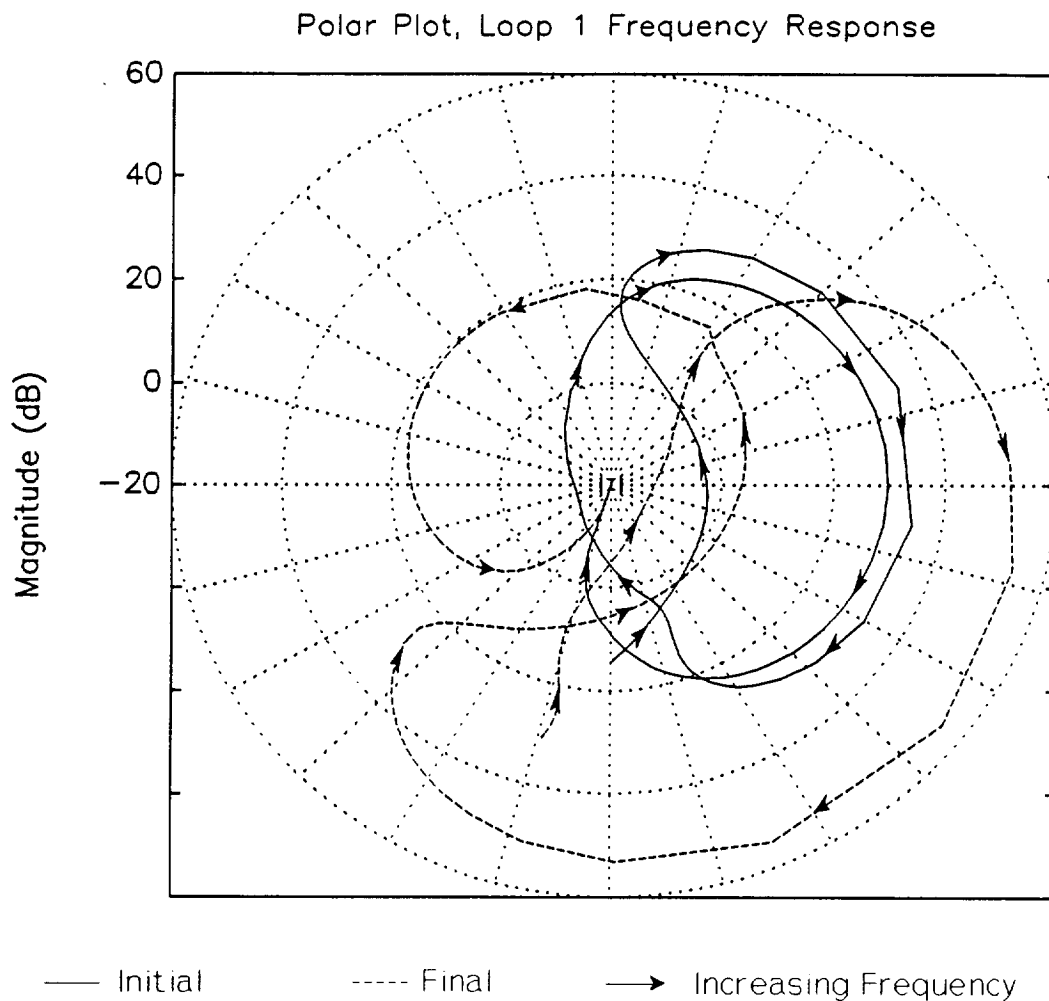


Figure 3.23: Polar Plot of the Compensated Open Loop System from control input $u(1)$ to measured output $y(1)$.

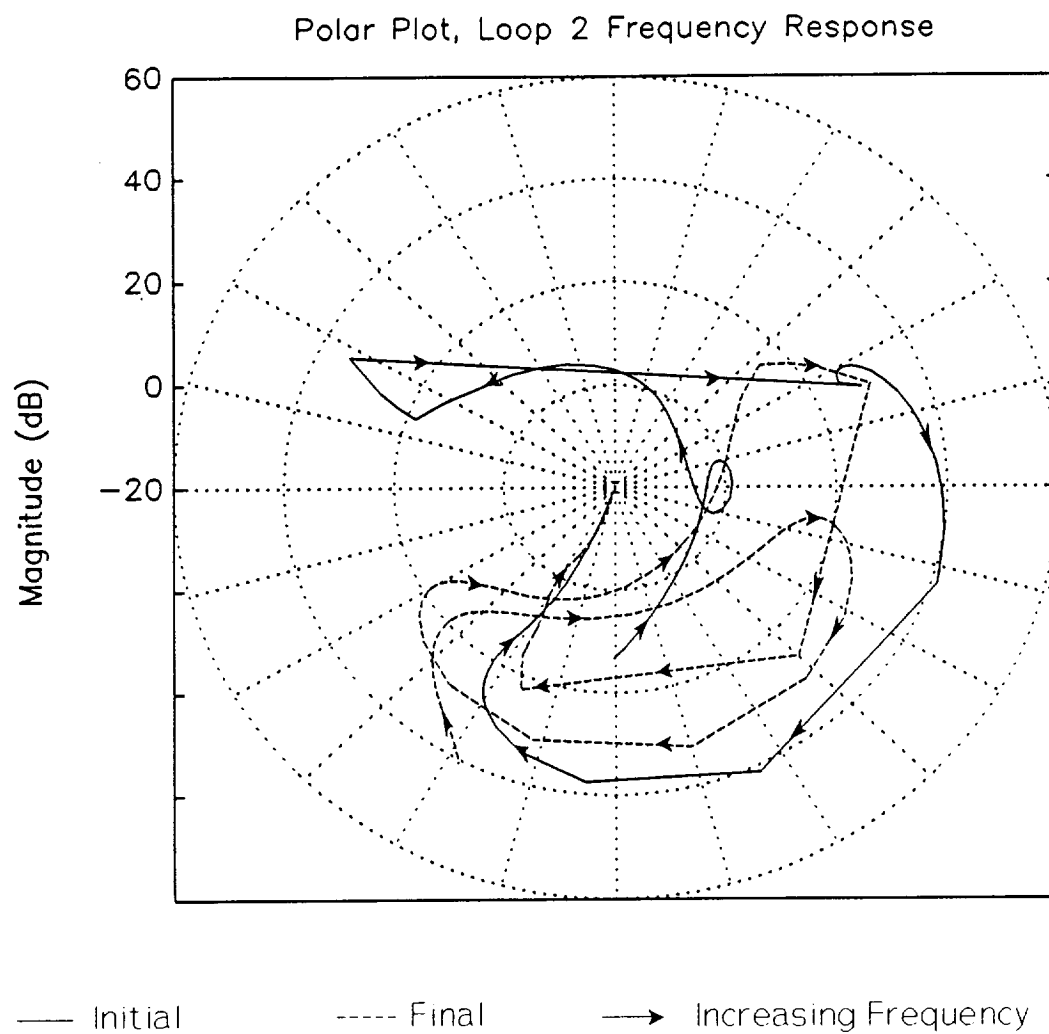


Figure 3.24: Polar Plot of the Compensated Open Loop System from control input $u(2)$ to measured output $y(2)$.

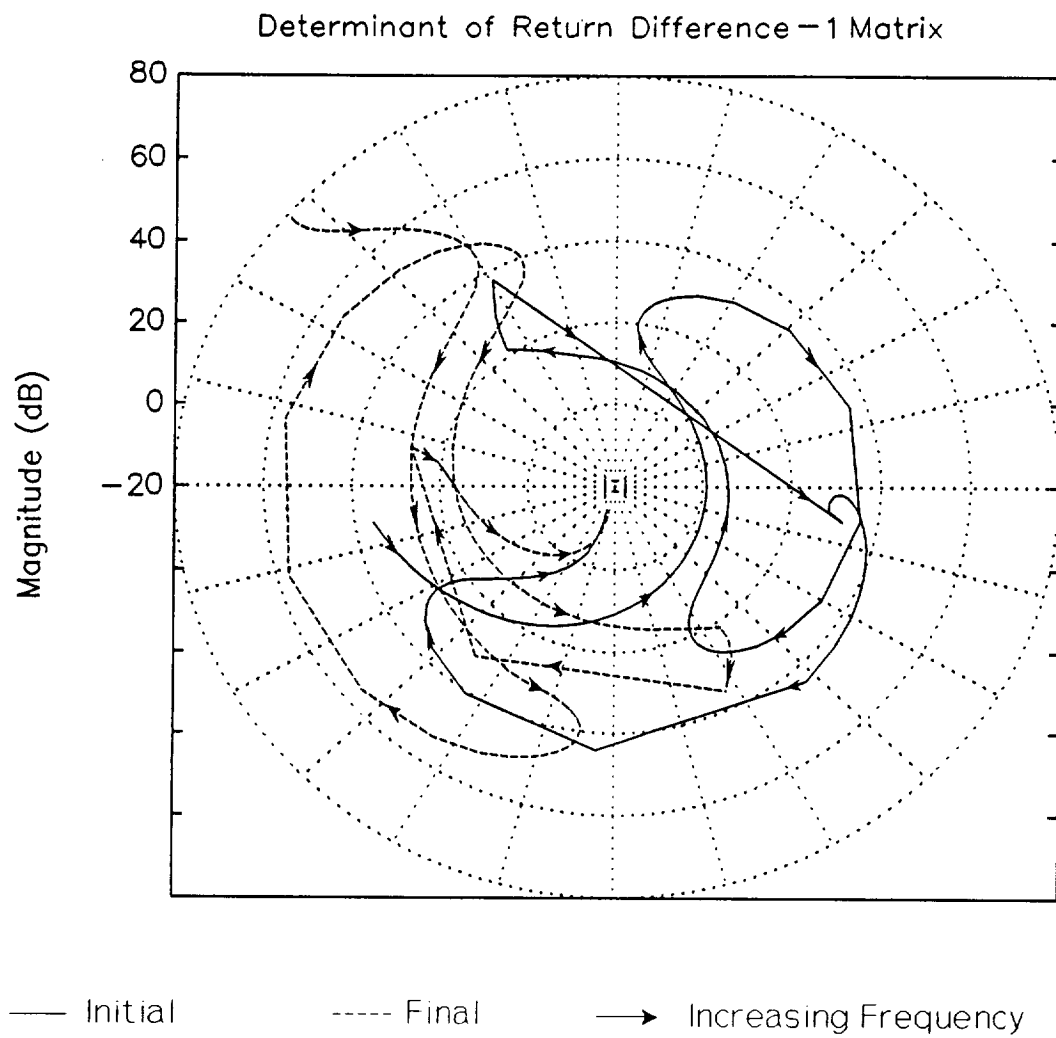


Figure 3.25: Polar Plots of Initial and Final Frequency Response Determinant of Return Difference - 1 Matrix.

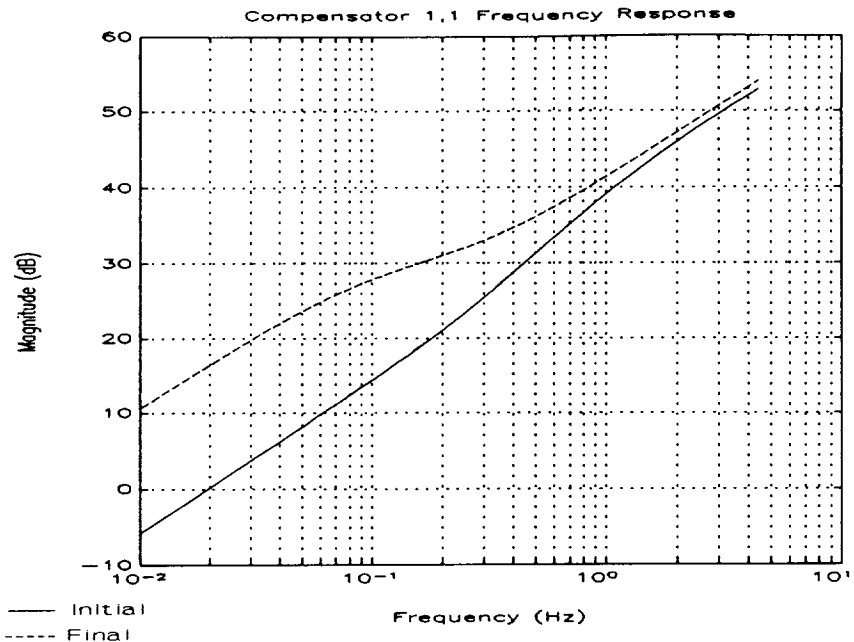


Figure 3.26: Magnitude Frequency Response of Compensator 1,1

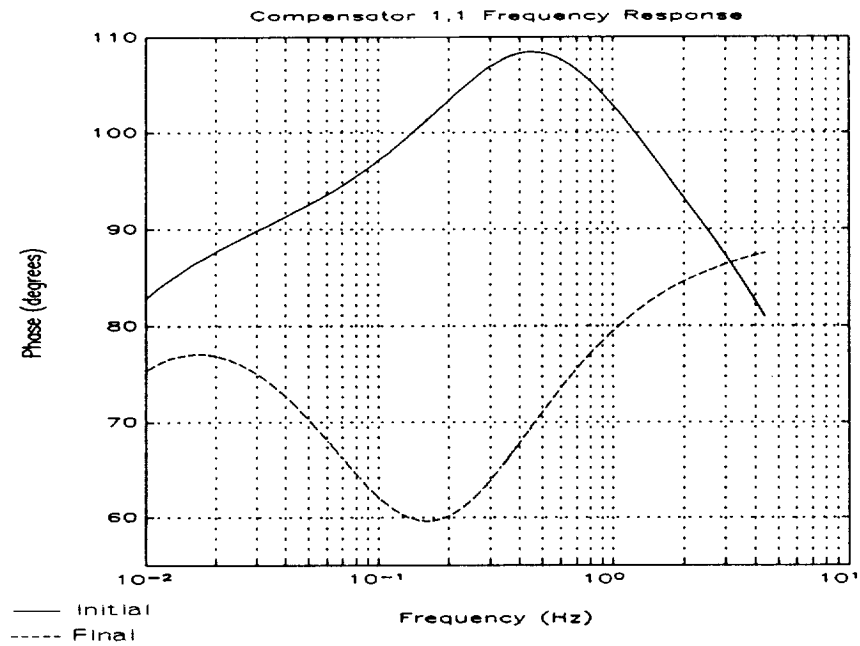


Figure 3.27: Phase Frequency Response of Compensator 1,1.

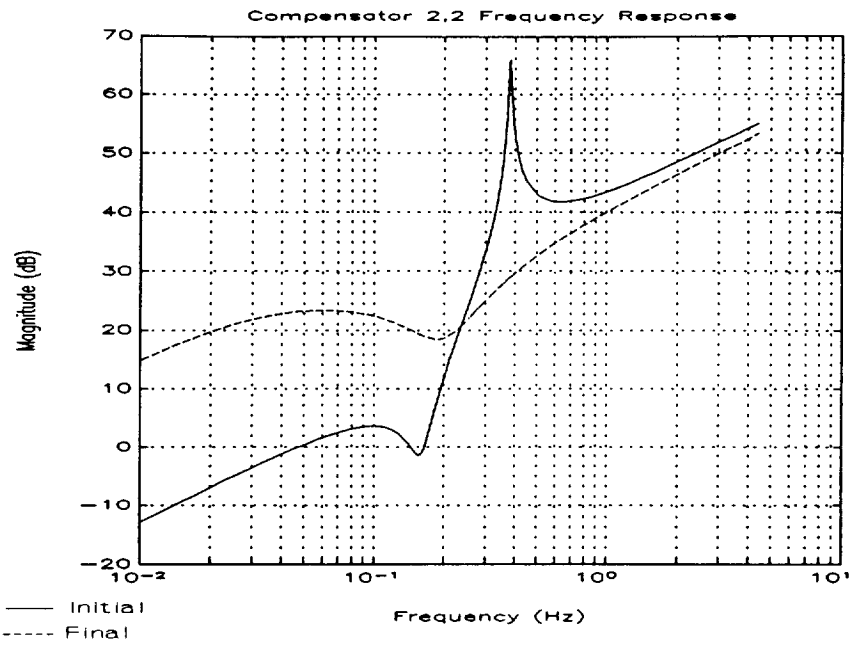


Figure 3.28: Magnitude Frequency Response of Compensator 2,2.

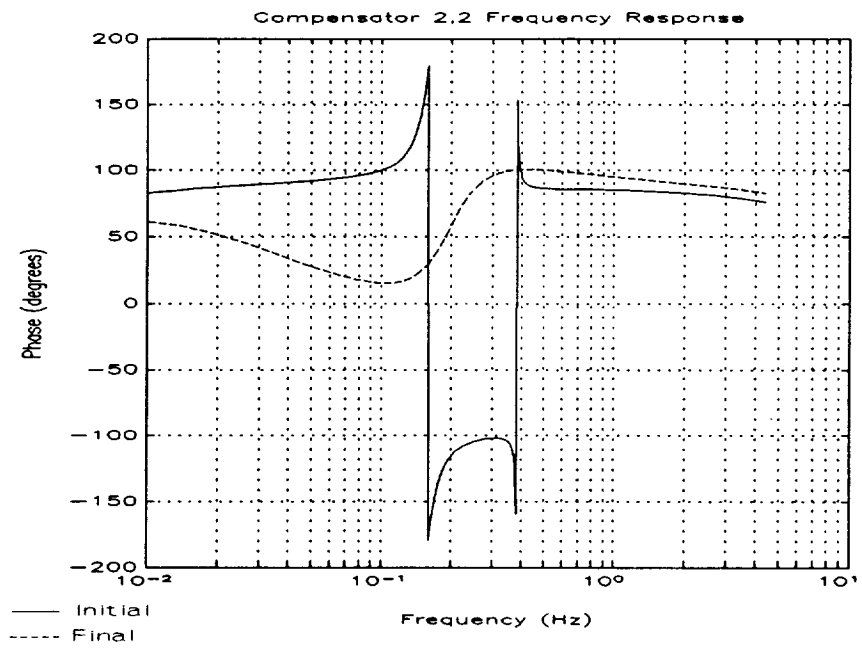


Figure 3.29: Phase Frequency Response of Compensator 2,2.

3.7.3 Pointing System Example (z-plane)

The following example involves a discretized version of the plant shown in the previous example. Due to the discretization of the data, which is done by using a sampler/zero-order-hold device at each system input, an inherent delay results, making the design more difficult to improve. Figures 3.30 to 3.37 show the open loop frequency responses of the uncompensated plant. The sampling period is 0.1 seconds. Notice the roll off as the frequencies approach the half sampling frequency.

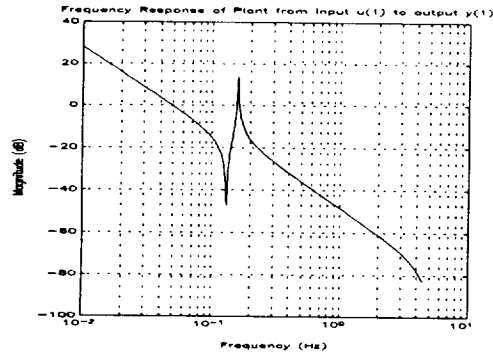


Figure 3.30: Mag. Frequency Response of Plant from $x(1)$ to $y(1)$

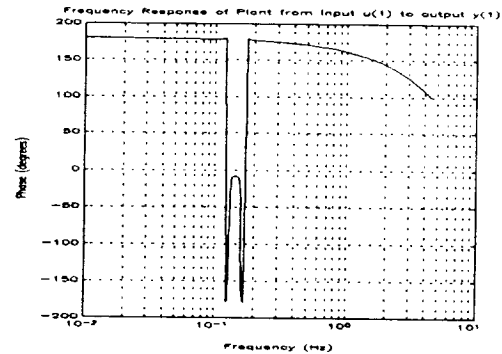


Figure 3.31: Phase Frequency Response of Plant from $x(1)$ to $y(1)$

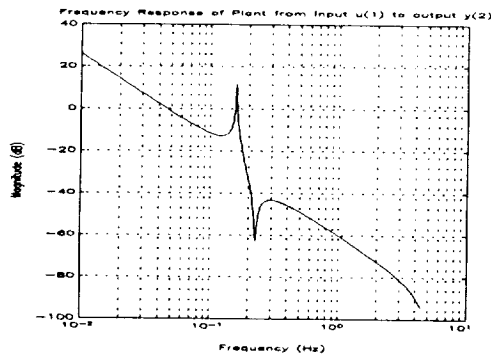


Figure 3.32: Mag. Frequency Response of Plant from $x(1)$ to $y(2)$

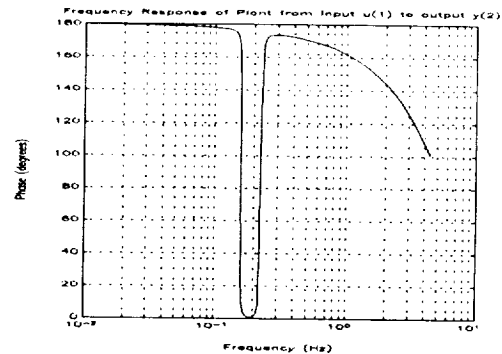


Figure 3.33: Phase Frequency Response of Plant from $x(1)$ to $y(2)$

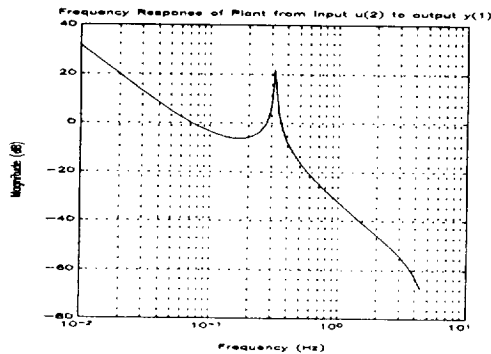


Figure 3.34: Mag. Frequency Response of Plant from $x(2)$ to $y(1)$

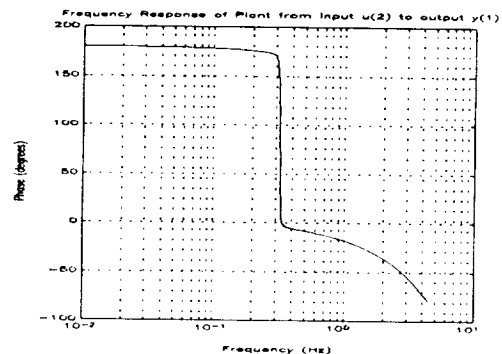


Figure 3.35: Phase Frequency Response of Plant from $x(2)$ to $y(1)$

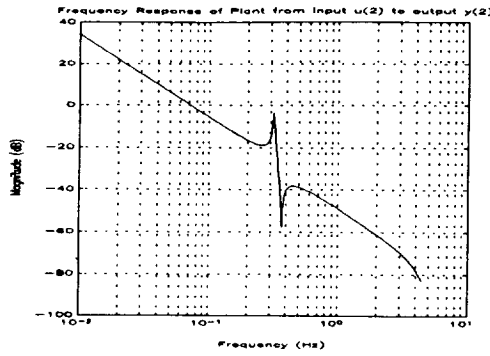


Figure 3.36: Mag. Frequency Response of Plant from $x(2)$ to $y(2)$

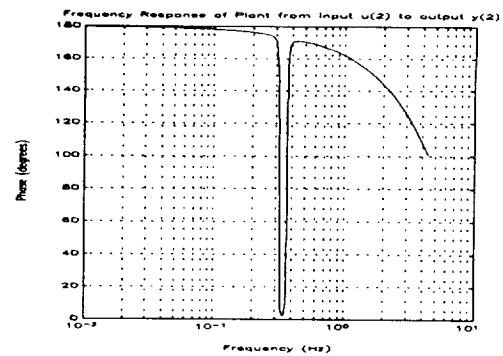


Figure 3.37: Phase Frequency Response of Plant from $x(2)$ to $y(2)$

All the types of specifications available in OUCIP were used in this example. All design specifications were achieved at the end of iteration 220.

Table 3.5 and Table 3.6 show the initial and the final factors of the compensator matrix elements. From Table 3.7 and Table 3.8 it is seen that considerable improvement to the damping ratios and DC gains has been accomplished. Also, the unsatisfied phase margin and attenuation level of loop 2 are now within the desired specifications.

Table 3.5 Initial Compensator Elements, z-plane Pointing Example

	Zeroth Order	First Order	Second Order
Numerator 1 Comp.(1,1)	-1948.98	1950.68	N/A
Denominator 1 Comp.(1,1)	0.703009	1.00000	N/A
Numerator 2 Comp.(1,1)	-1.50000	1.85505	N/A
Denominator 2 Comp.(1,1)	-0.644949	1.00000	N/A
Numerator 1 Comp.(2,2)	-1948.98	1950.68	N/A
Denominator 1 Comp.(2,2)	0.703009	1.00000	N/A
Numerator 2 Comp.(2,2)	5.20662	-10.4657	5.31163
Denominator 2 Comp.(2,2)	0.995451	-1.94290	1.00000

Table 3.6 Final (iteration 220) Compensator Elements, z-plane Pointing Example

	Zeroth Order	First Order	Second Order
Numerator 1 Comp.(1,1)	-2293.00	2295.01	N/A
Denominator 1 Comp.(1,1)	1.00223	1.00000	N/A
Numerator 2 Comp.(1,1)	-1.87765	2.05740	N/A
Denominator 2 Comp.(1,1)	-0.977165	1.00000	N/A
Numerator 1 Comp.(2,2)	-2032.19	2033.96	N/A
Denominator 1 Comp.(2,2)	0.774639	1.00000	N/A
Numerator 2 Comp.(2,2)	3.99664	-8.39197	4.48299
Denominator 2 Comp.(2,2)	0.546768	-1.54189	1.00000

Table 3.7 Initial Objective Function Values for the z-plane Pointing Example

Iteration	Type	Frequency (Hz)	Desired	Current
0	G1	0.2825	≥ 0.5000	0.8243
0	G1	3.881	≥ 0.5000	0.9095
0	P1	0.06362	$\geq 50.0^\circ$	140.1°
0	P1	0.1303	$\geq 50.0^\circ$	130.1°
0	P1	0.2349	$\geq 50.0^\circ$	97.06°
0	P1	0.3041	$\geq 50.0^\circ$	75.96°
0	P1	0.7013	$\geq 50.0^\circ$	56.37°
0	A1	2.059	≤ 0.2000	0.1668
0	A1	4.405	≤ 0.2000	0.07974
0	G2	3.881	≥ 0.5000	0.8851
0	P2	0.04497	$\geq 50.0^\circ$	125.1°
0	P2	0.1238	$\geq 50.0^\circ$	168.8°
0	P2	0.1444	$\geq 50.0^\circ$	164.2°
0	P2	0.2182	$\geq 50.0^\circ$	114.8°
0	P2	0.7961	$\geq 50.0^\circ$	44.37°
0	A2	2.059	≤ 0.2000	0.2170
0	A2	4.405	≤ 0.2000	0.09469
0	Z22 ζ	0.1592	≥ 0.3000	0.09984
0	P22 ζ	0.3661	≥ 0.3000	0.00991
0	DC11	0.0000	≥ 0.5000	0.07079
0	DC22	0.0000	≥ 0.5000	0.03162
0	DR11	N/A	≤ 0.0070	0.00462
0	DR22	N/A	≤ 0.0070	0.00349
0	DP11	0.0100	≤ 0.0070	0.01880
0	DP11	0.2885	≤ 0.0070	0.00958
0	DP11	0.3107	≤ 0.0070	0.00958
0	DP22	0.0000	≤ 0.0070	0.02252

Table 3.8 Final Objective Function Values for the z-plane Pointing Example

Iteration	Type	Frequency (Hz)	Desired	Current
220	G1	0.3460	≥ 0.5000	3.543
220	P1	0.05586	$\geq 50.0^\circ$	73.62°
220	P1	0.06740	$\geq 50.0^\circ$	157.50°
220	P1	0.61780	$\geq 50.0^\circ$	50.00°
220	A1	2.059	≤ 0.2000	0.1984
220	A1	4.405	≤ 0.2000	0.1002
220	G2	3.881	≥ 0.5000	0.8839
220	P2	0.06092	$\geq 50.0^\circ$	80.74°
220	P2	0.06740	$\geq 50.0^\circ$	131.0°
220	P2	0.2026	$\geq 50.0^\circ$	58.5°
220	P2	0.2877	$\geq 50.0^\circ$	159.0°
220	P2	0.4500	$\geq 50.0^\circ$	71.92°
220	A2	2.059	≤ 0.2000	0.19580
220	A2	4.405	≤ 0.2000	0.09709
220	Z22 ζ	0.2218	≥ 0.3000	≥ 0.3000
220	P22 ζ	0.1054	≥ 0.3000	≥ 0.3000
220	DC11	0.0000	≥ 0.5000	≥ 0.5000
220	DC22	0.0000	≥ 0.5000	≥ 0.5000
220	DR11	N/A	≤ 0.0070	0.00374
220	DR22	N/A	≤ 0.0070	0.00207
220	DP11	0.01	≤ 0.0070	≤ 0.0070
220	DP11	Unavailable	≤ 0.0070	≤ 0.0070
220	DP11	Unavailable	≤ 0.0070	≤ 0.0070
220	DP22	0.01	≤ 0.0070	≤ 0.0070

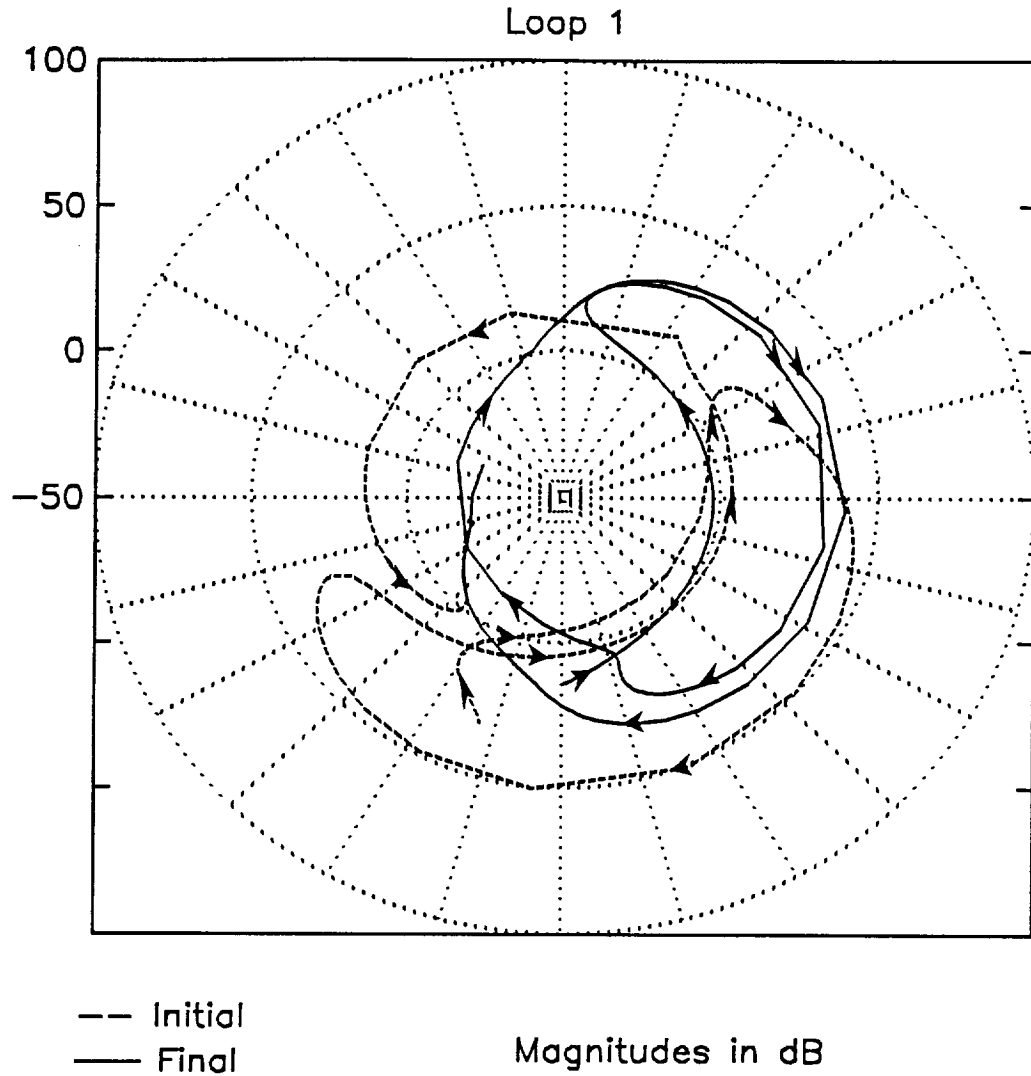


Figure 3.38: Polar plot of the Compensated Open Loop System from control input $u(1)$ to measured output $y(1)$

Figure 3.38 and Figure 3.39 show the polar plots of the compensated open loop systems. From these it can be determined that the compensated open loop system 1 has lost loop integrity as in the design in the s -plane example. Similarly the closed loop system is assumed to have remained stable by examining the determinant of the return difference - 1 matrix shown in Figure 3.40.

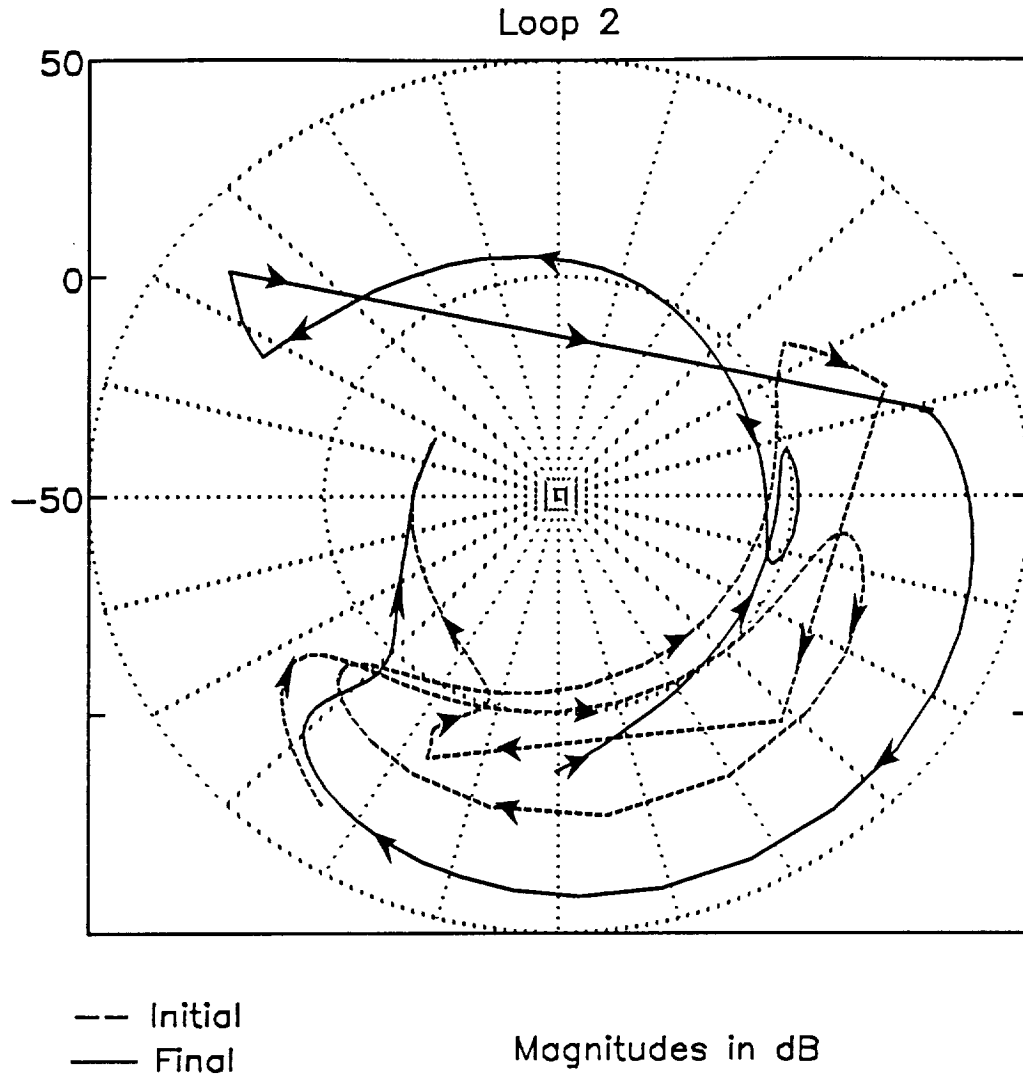


Figure 3.39: Polar Plot of the Compensated Open Loop System from control input $u(2)$ to measured output $y(2)$

Figures 3.41 to 3.44 show the magnitude and phase frequency responses of the compensator. Notice the improvement of the low frequency gain. This is due to the increase of the compensator DC gains. ? shows the improvement of compensator damping ratios by the smoothing out of the peaks and notches of the frequency response.

Finally, Figures 3.45 and 3.46 show that the peaks of the magnitude frequency responses from disturbance 1 to measured output 1 and from disturbance 2 to measured output 2 did not satisfy the specifications with the initial compensator, but are below the desired level with the final compensator.

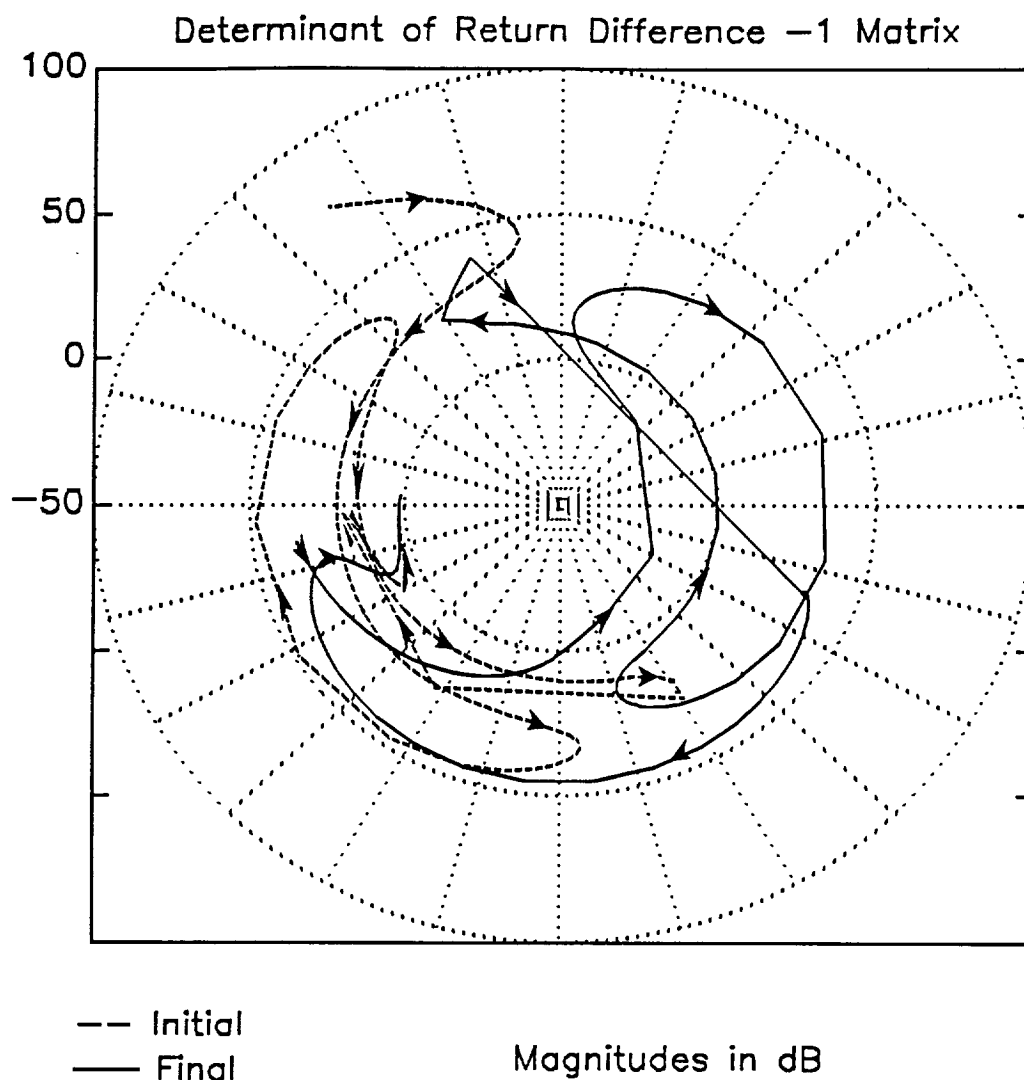


Figure 3.40: Polar Plot of the Frequency Response of the Determinant of the Return Difference - 1 Matrix.

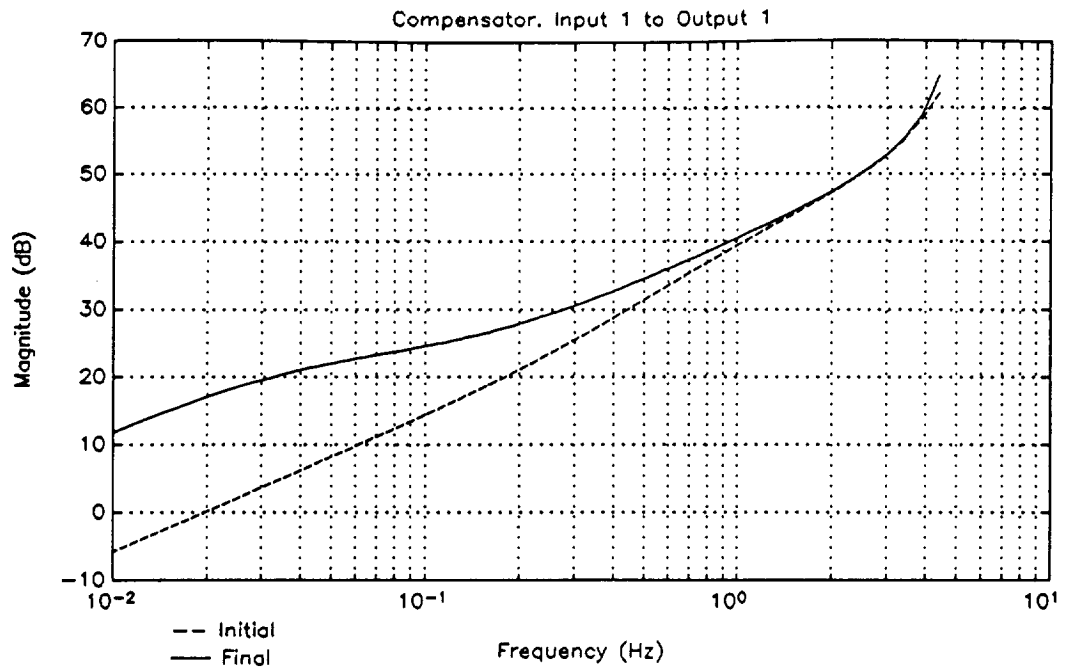


Figure 3.41: Magnitude Frequency Response of the Compensator 1,1

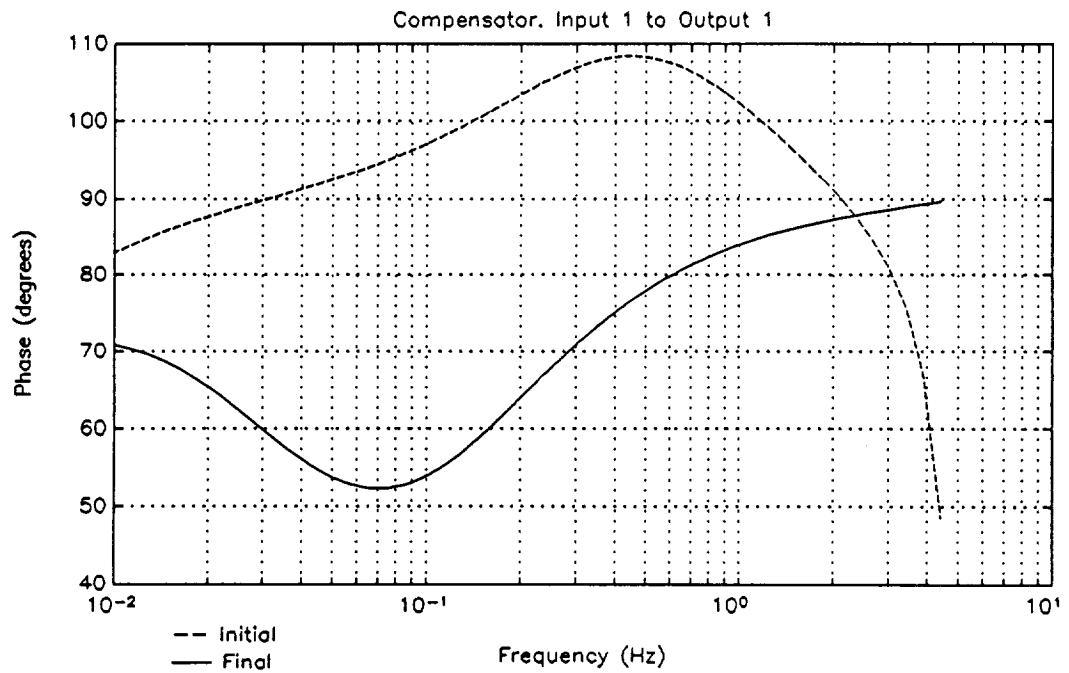


Figure 3.42: Phase Frequency Response of the Compensator 1,1.

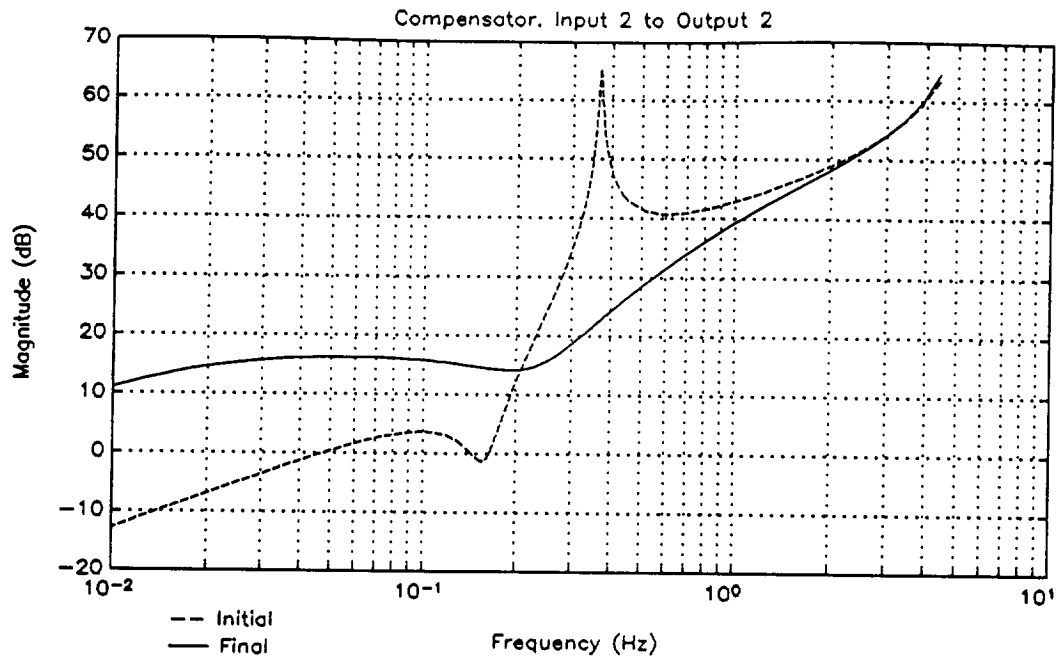


Figure 3.43: Magnitude Frequency Response of the Compensator 2,2

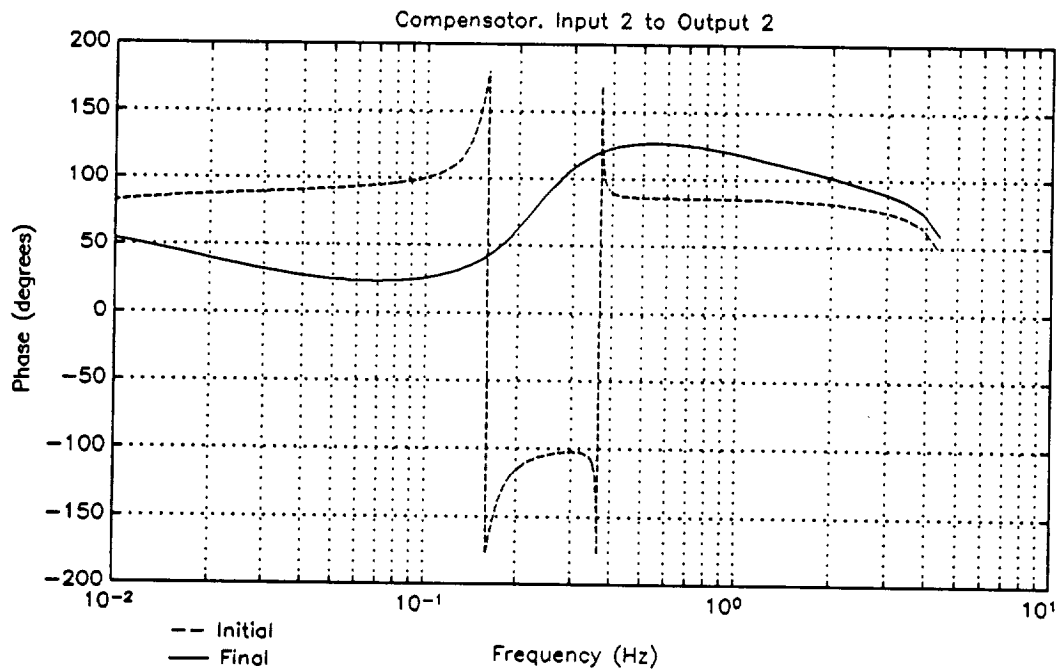


Figure 3.44: Phase Frequency Response of the Compensator 2,2

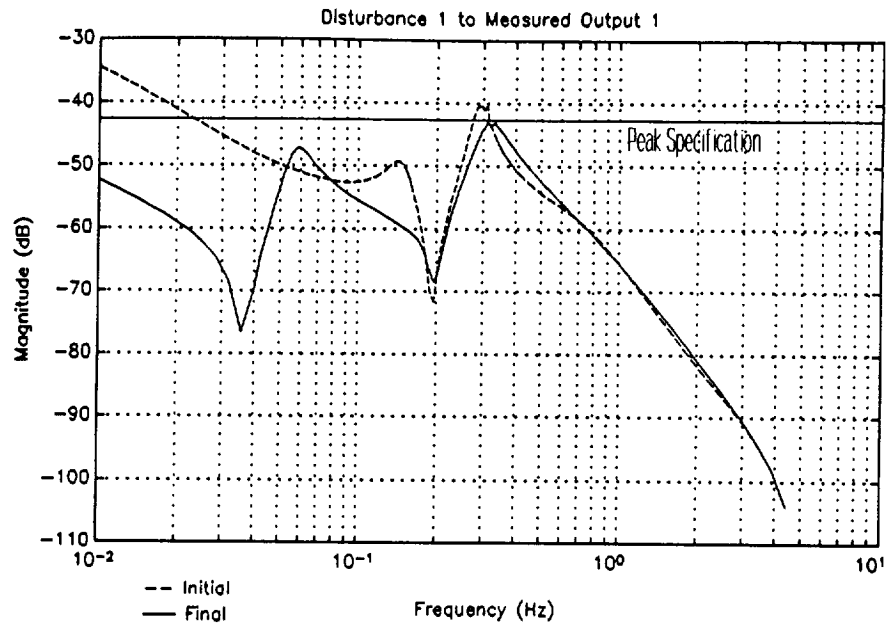


Figure 3.45 Magnitude Frequency Response from Disturbance 1 to Output 1 for the z-plane Pointing Example.

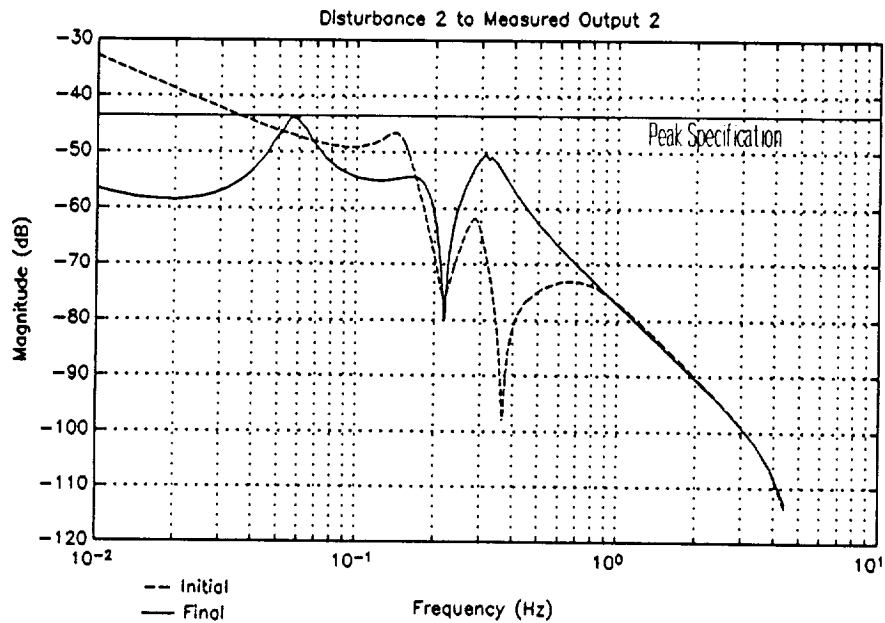


Figure 3.46 Magnitude Frequency Response from Disturbance 2 to Output 2 for the z-plane Pointing Example

3.7.4 Example 2: Space Shuttle (w-plane)

This is an example of the Yaw/Roll Ascent Flight Control System for the Space Shuttle. The system has two control inputs and four measured outputs.

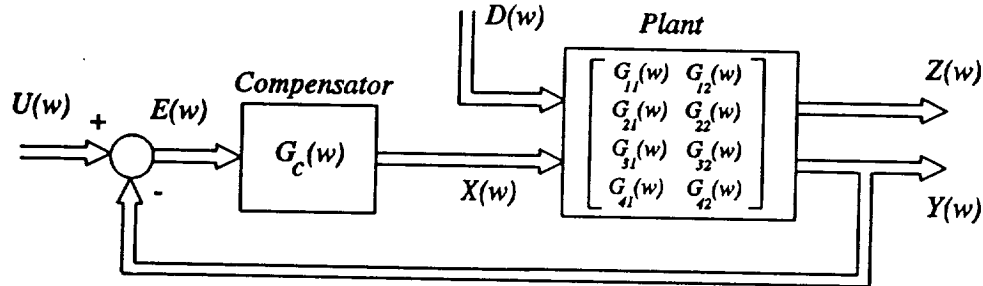


Figure 3.47: Shuttle Yaw / Roll Control System Block Diagram

The plant is described by 26 discrete frequency response data points for each of the 8 open loop systems. Many of the modes of this system are lightly damped. This design was initially performed in the w-plane, with a sampling time of 0.04 seconds. Figures 3.46 to 3.53 show the open loop frequency responses of the uncompensated plant.

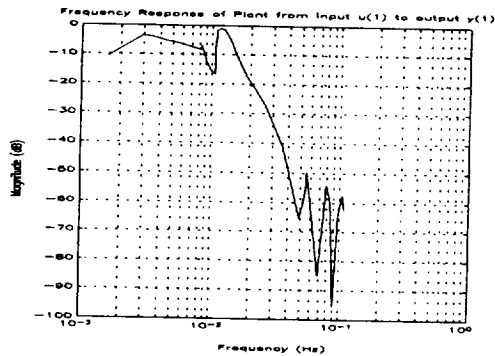


Figure 3.48: Mag. Frequency Response of Plant from x(1) to y(1)

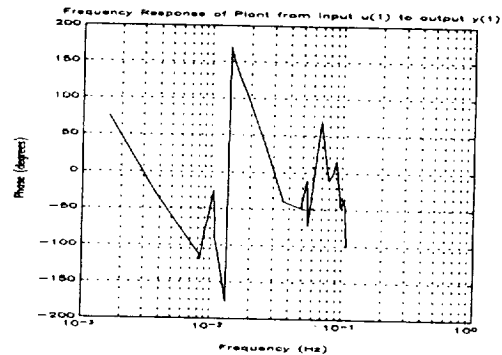


Figure 3.49: Phase Frequency Response of Plant from x(1) to y(1)

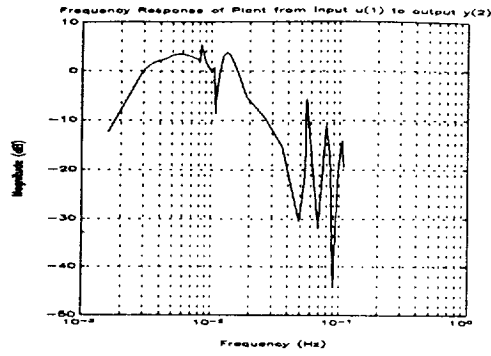


Figure 3.50: Magnitude Frequency Response of Plant from $x(1)$ to $y(2)$

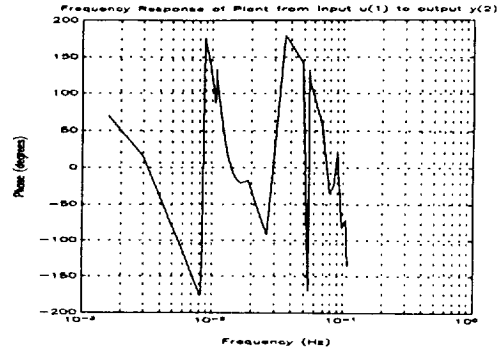


Figure 3.51: Phase Frequency Response of Plant from $x(1)$ to $y(2)$

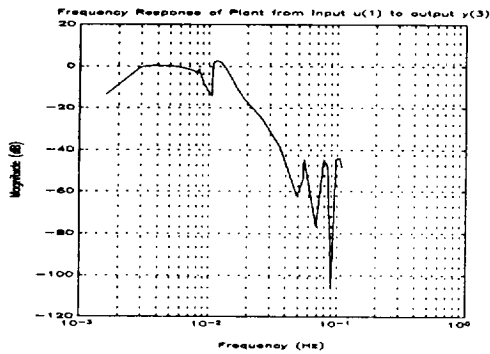


Figure 3.52: Magnitude Frequency Response of Plant from $x(1)$ to $y(3)$

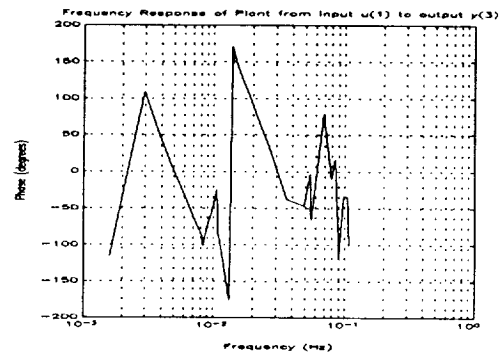


Figure 3.53: Phase Frequency Response of Plant from $x(1)$ to $y(3)$

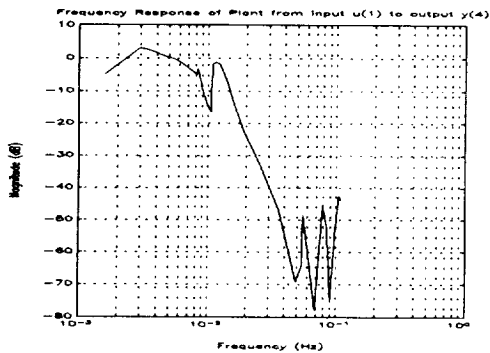


Figure 3.54: Magnitude Frequency Response of Plant from $x(1)$ to $y(4)$

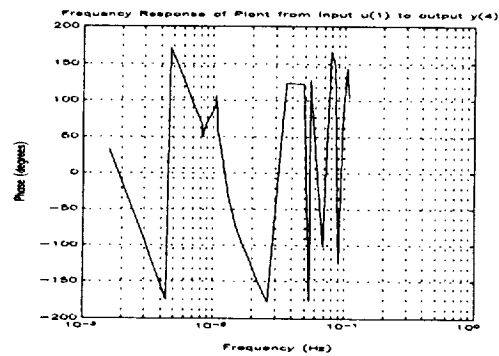


Figure 3.55: Phase Frequency Response of Plant from $x(1)$ to $y(4)$

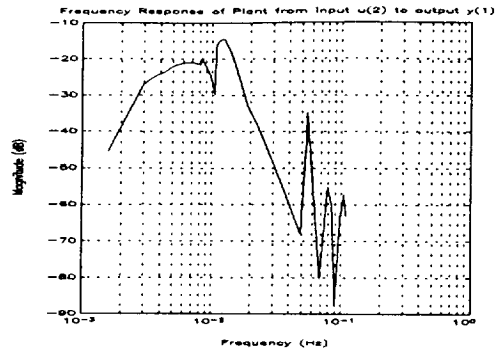


Figure 3.56: Magnitude Frequency Response of Plant from $x(2)$ to $y(1)$

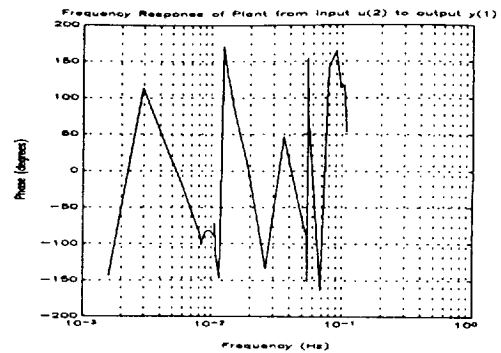


Figure 3.57: Phase Frequency Response of Plant from $x(2)$ to $y(2)$

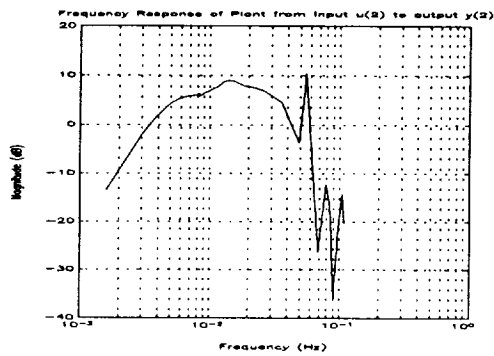


Figure 3.58: Magnitude Frequency Response of Plant from $x(2)$ to $y(2)$

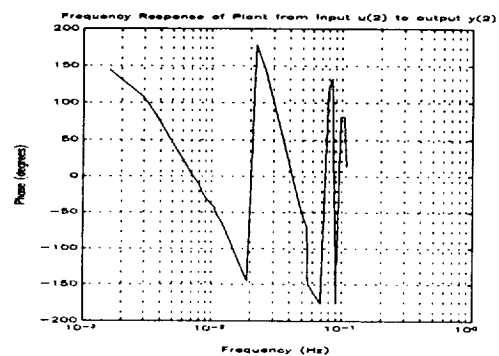


Figure 3.59: Phase Frequency Response of Plant from $x(2)$ to $y(2)$

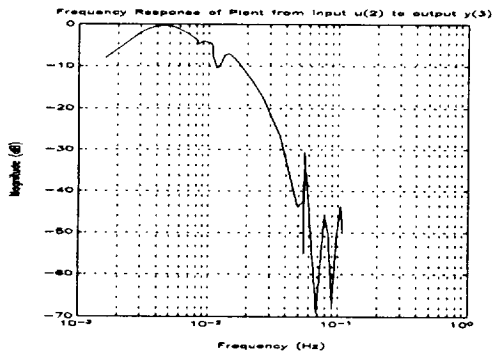


Figure 3.60: Magnitude Frequency Response of Plant from $x(2)$ to $y(3)$

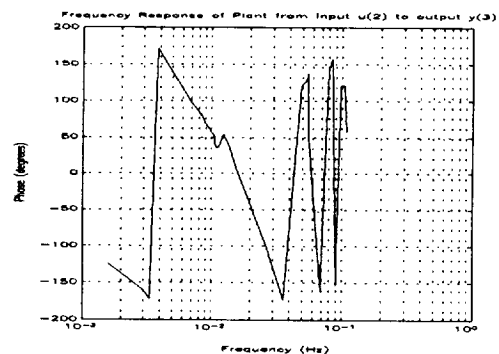


Figure 3.61: Phase Frequency Response of Plant from $x(2)$ to $y(3)$

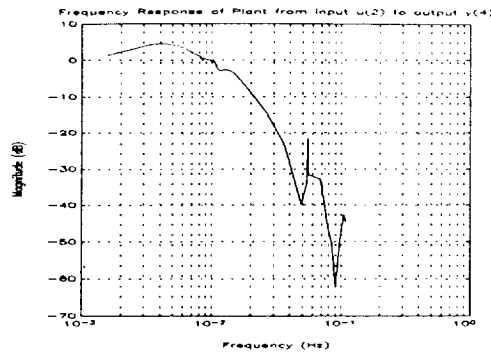


Figure 3.62: Magnitude Frequency Response of Plant from $x(2)$ to $y(4)$

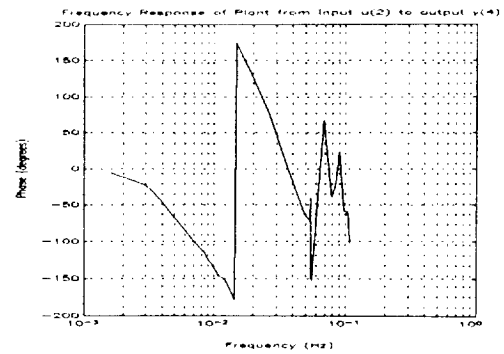


Figure 3.63: Phase Frequency Response of Plant from $x(2)$ to $y(4)$

A satisfactory design for this example was accomplished in stages by using the ACTIVATE menu to turn on certain design specifications. Initially only the relative stability specifications were activated. The DC gains were set to stay constant (initial) values. During this time, some compensator damping ratios were degraded. This execution finished at iteration 39 when all relative stability specifications were achieved. Next, the compensator damping ratios were added to the specifications to be improved. These were allowed to improve until all damping ratios of the compensator poles were satisfied ($\zeta_{\text{DEN}}\text{'s} \geq 0.499$). At this point the DC gains were activated along with previously activated specifications and execution continued until DC gains were larger than 0.7 and damping ratios of the compensator zeros increased to 0.4. This execution ended after iteration 136 when all current specification levels were satisfied. The desired specification values were then increased. The DC gains were increased to 0.8 and both zero and pole damping ratios were increased to 0.5. This execution finished at iteration 184 when all design specifications were achieved. At this point the compensator DC gain specifications were raised to 0.9. This design improvement was accomplished at the end iteration 190. Many other executions were attempted with different combinations of active specifications, but this series of phases seemed to work the best. The damping ratios were unable to be raised much above 0.5.

Tables 3.9 and 3.10 give the initial and final compensator element coefficients in ascending order. Tables 3.11 and 3.12 show the initial and final specification values.

Figures 3.64 and 3.65 show the frequency response of the broken loop systems 1 and 2 respectively. These plots show the improvement of the phase margins and gain margins of each system. From Figure 3.66 the stability of the system can be seen to have remain unchanged from the initial iteration to the final design.

Figures 3.67 to 3.74 show the magnitude and phase frequency response plots of the compensators. Notice the smoothing out of the notches caused by increasing the damping ratios.

Table 3.9 Initial Compensator Elements (w-plane)

	Zeroth Order	First Order	Second Order
Numerator 1 Comp.(1,1)	0.01000	1.00000	N/A
Denominator 1 Comp.(1,1)	0.01330	1.00000	N/A
Numerator 2 Comp.(1,1)	1.00000	1.00000	8.16300
Denominator 2 Comp.(1,1)	1.00000	4.34800	18.9030
Numerator 3 Comp.(1,1)	1.00000	2.28600	2.01400
Denominator 3 Comp.(1,1)	1.00000	4.34800	18.9030
Numerator 4 Comp.(1,1)	1.00000	1.20000	4.44400
Denominator 4 Comp.(1,1)	1.00000	2.40000	4.00000
Numerator 5 Comp.(1,1)	1.00000	1.00000	8.16300
Denominator 5 Comp.(1,1)	1.00000	4.34800	18.9030
Numerator 1 Comp.(1,2)	1.00000	-1.00000	N/A
Denominator 1 Comp.(1,2)	1.00000	50.0074	N/A
Numerator 2 Comp.(1,2)	0.01000	1.00000	N/A
Denominator 2 Comp.(1,2)	0.01330	1.00000	N/A
Numerator 3 Comp.(1,2)	1.00000	0.57140	8.16300
Denominator 3 Comp.(1,2)	1.00000	5.21700	18.9030
Numerator 4 Comp.(1,2)	1.00000	0.64520	10.4100
Denominator 4 Comp.(1,2)	1.00000	6.00000	25.0000
Numerator 5 Comp.(1,2)	1.00000	0.57140	2.04100
Denominator 5 Comp.(1,2)	1.00000	5.33300	11.1111
Numerator 6 Comp.(1,2)	1.00000	1.20000	4.44400
Denominator 6 Comp.(1,2)	1.00000	2.40000	4.00000
Numerator 1 Comp.(1,3)	0.00000	51.0074	N/A
Denominator 1 Comp.(1,3)	1.00000	50.0074	N/A
Numerator 2 Comp.(1,3)	0.01000	1.00000	N/A
Denominator 2 Comp.(1,3)	0.01330	1.00000	N/A

Numerator 3 Comp.(1,3)	1.00000	0.57140	8.16300
Denominator 3 Comp.(1,3)	1.00000	5.21700	18.9030
Numerator 4 Comp.(1,3)	1.00000	0.64520	10.4100
Denominator 4 Comp.(1,3)	1.00000	6.00000	25.00000
Numerator 5 Comp.(1,3)	1.00000	0.57140	2.04100
Denominator 5 Comp.(1,3)	1.00000	5.33300	11.1110
Numerator 6 Comp.(1,3)	1.00000	1.20000	4.44400
Denominator 6 Comp.(1,3)	1.00000	2.40000	4.00000
Numerator 1 Comp.(2,4)	0.03000	1.00000	N/A
Denominator 1 Comp.(2,4)	0.04000	1.00000	N/A
Numerator 2 Comp.(2,4)	1.00000	0.85710	8.16300
Denominator 2 Comp.(2,4)	1.00000	4.34800	18.9030
Numerator 3 Comp.(2,4)	1.00000	1.00000	6.25000
Denominator 3 Comp.(2,4)	1.00000	4.34800	18.9030
Numerator 4 Comp.(2,4)	1.00000	2.00000	1.56300
Denominator 4 Comp.(2,4)	1.00000	3.33300	11.1110
Numerator 5 Comp.(2,4)	1.00000	1.20000	4.44400
Denominator 5 Comp.(2,4)	1.00000	2.40000	4.00000

Table 3.10: Final (iteration 190) Compensator Elements (w-plane)

	Zeroth Order	First Order	Second Order
Numerator 1 Comp.(1,1)	0.0114689	0.854976	N/A
Denominator 1 Comp.(1,1)	0.0121452	1.12287	N/A
Numerator 2 Comp.(1,1)	1.00002	2.87901	8.15254
Denominator 2 Comp.(1,1)	0.999985	4.34426	18.8722
Numerator 3 Comp.(1,1)	1.00002	2.67403	2.03328
Denominator 3 Comp.(1,1)	0.999985	4.34426	18.8722
Numerator 4 Comp.(1,1)	1.00002	2.11937	4.45197
Denominator 4 Comp.(1,1)	0.999985	2.00405	3.98085
Numerator 5 Comp.(1,1)	1.00002	2.87901	8.15254
Denominator 5 Comp.(1,1)	0.999985	4.34426	18.8722
Numerator 1 Comp.(1,2)	1.00004	-0.922847	N/A
Denominator 1 Comp.(1,2)	0.999961	50.0110	N/A
Numerator 2 Comp.(1,2)	0.0134783	0.217520	N/A
Denominator 2 Comp.(1,2)	0.010130	1.52257	N/A
Numerator 3 Comp.(1,2)	1.00004	2.88835	8.11781
Denominator 3 Comp.(1,2)	0.999961	5.13476	18.9016
Numerator 4 Comp.(1,2)	1.00004	3.23073	10.3597
Denominator 4 Comp.(1,2)	0.999961	5.91867	24.9981
Numerator 5 Comp.(1,2)	1.00004	1.42675	2.02449
Denominator 5 Comp.(1,2)	0.999961	5.25500	11.1098
Numerator 6 Comp.(1,2)	1.00004	2.11969	4.42663
Denominator 6 Comp.(1,2)	0.999961	2.31828	4.00029
Numerator 1 Comp.(1,3)	0.00000	51.0047	N/A
Denominator 1 Comp.(1,3)	0.999961	50.0193	N/A
Numerator 2 Comp.(1,3)	0.0124783	0.308931	N/A
Denominator 2 Comp.(1,3)	0.0101301	1.41465	N/A

Numerator 3 Comp.(1,3)	1.00004	2.86469	9.13119
Denominator 3 Comp.(1,3)	0.999961	5.15289	18.8902
Numerator 4 Comp.(1,3)	1.00004	3.26889	10.3721
Denominator 4 Comp.(1,3)	0.999961	5.94790	24.9870
Numerator 5 Comp.(1,3)	1.00004	1.46337	2.03603
Denominator 5 Comp.(1,3)	0.999961	5.26898	11.0989
Numerator 6 Comp.(1,3)	1.00004	2.11264	4.43917
Denominator 6 Comp.(1,3)	0.999961	2.29927	3.98871
Numerator 1 Comp.(2,4)	0.033576	0.795842	N/A
Denominator 1 Comp.(2,4)	0.0371472	0.739216	N/A
Numerator 2 Comp.(2,4)	1.00011	2.88687	8.17731
Denominator 2 Comp.(2,4)	0.999889	4.35432	18.8474
Numerator 3 Comp.(2,4)	1.00011	2.51097	6.27352
Denominator 3 Comp.(2,4)	0.999889	4.35432	18.8474
Numerator 4 Comp.(2,4)	1.00011	2.33121	1.60744
Denominator 4 Comp.(2,4)	0.999889	3.34105	11.0565
Numerator 5 Comp.(2,4)	1.00011	2.12802	4.47811
Denominator 5 Comp.(2,4)	0.999889	2.07043	3.95442

Table 3.11: Initial Objective Function Values for the w-plane Shuttle Example

Iteration	Type	Frequency (Hz)	Desired	Current
0	G1	0.01425	≥ 0.6000	0.4163
0	P1	0.003822	$\geq 45.0^\circ$	53.74°
0	P1	0.009590	$\geq 45.0^\circ$	127.5°
0	P1	0.01081	$\geq 45.0^\circ$	24.21°
0	G2	0.01425	≥ 0.6000	0.5941
0	P2	0.005103	$\geq 45.0^\circ$	36.50°
0	Z11 ζ	0.05570	≥ 0.5000	0.1750
0	Z11 ζ	0.07550	≥ 0.5000	0.2846
0	Z11 ζ	0.05570	≥ 0.5000	0.1750
0	Z11 ζ	0.11210	≥ 0.5000	0.8054
0	P11 ζ	0.03661	≥ 0.5000	0.5000
0	P11 ζ	0.03661	≥ 0.5000	0.5000
0	P11 ζ	0.07958	≥ 0.5000	0.6000
0	P11 ζ	0.03661	≥ 0.5000	0.5000
0	Z12 ζ	0.05570	≥ 0.5000	0.1000
0	Z12 ζ	0.04933	≥ 0.5000	0.1000
0	Z12 ζ	0.11140	≥ 0.5000	0.2000
0	Z12 ζ	0.07550	≥ 0.5000	0.2846
0	P12 ζ	0.03661	≥ 0.5000	0.6000
0	P12 ζ	0.03183	≥ 0.5000	0.6000
0	P12 ζ	0.04775	≥ 0.5000	0.7999
0	P12 ζ	0.07958	≥ 0.5000	0.6000
0	Z13 ζ	0.05570	≥ 0.5000	0.1000
0	Z13 ζ	0.04933	≥ 0.5000	0.1000
0	Z13 ζ	0.11140	≥ 0.5000	0.2000
0	Z13 ζ	0.07550	≥ 0.5000	0.2846

0	P13ζ	0.03361	≥ 0.5000	0.6000
0	P13ζ	0.03183	≥ 0.5000	0.6000
0	P13ζ	0.04775	≥ 0.5000	0.7999
0	P13ζ	0.07958	≥ 0.5000	0.6000
0	Z24ζ	0.05570	≥ 0.5000	0.1500
0	Z24ζ	0.06366	≥ 0.5000	0.2000
0	Z24ζ	0.07550	≥ 0.5000	0.2846
0	Z24ζ	0.12730	≥ 0.5000	0.7998
0	P24ζ	0.03661	≥ 0.5000	0.5000
0	P24ζ	0.03661	≥ 0.5000	0.5000
0	P24ζ	0.04775	≥ 0.5000	0.4999
0	P24ζ	0.07958	≥ 0.5000	0.6000
0	DC11	0.00000	≥ 0.9000	0.7519
0	DC12	0.00000	≥ 0.9000	0.5602
0	DC13	0.00000	≥ 0.9000	0.5602
0	DC24	0.00000	≥ 0.9000	0.7500

Table 3.12: Final Objective Function Values for the w-plane Shuttle Example.

Iteration	Type	Frequency (Hz)	Desired	Current
190	G1	0.01879	≥ 0.6000	0.7819
190	P1	0.003136	$\geq 45.00^\circ$	46.05°
190	P1	0.01050	$\geq 45.00^\circ$	131.7°
190	P1	0.01064	$\geq 45.00^\circ$	52.90°
190	G2	0.01879	≥ 0.6000	0.6340
190	P2	0.00606	$\geq 45.00^\circ$	46.89°
190	Z11 ζ	0.05574	≥ 0.5000	0.5044
190	Z11 ζ	0.11160	≥ 0.5000	0.9376
190	Z11 ζ	0.07543	≥ 0.5000	0.5026
190	Z11 ζ	0.05574	≥ 0.5000	0.5044
190	P11 ζ	0.03664	≥ 0.5000	0.5001
190	P11 ζ	0.03664	≥ 0.5000	0.5001
190	P11 ζ	0.07977	≥ 0.5000	0.5026
190	P11 ζ	0.03664	≥ 0.5000	0.5001
190	Z12 ζ	0.05586	≥ 0.5000	0.5071
190	Z12 ζ	0.04945	≥ 0.5000	0.5020
190	Z12 ζ	0.11190	≥ 0.5000	0.5022
190	Z12 ζ	0.07565	≥ 0.5000	0.5041
190	P12 ζ	0.03661	≥ 0.5000	0.5906
190	P12 ζ	0.03183	≥ 0.5000	0.5920
190	P12 ζ	0.04775	≥ 0.5000	0.7883
190	P12 ζ	0.07957	≥ 0.5000	0.5800
190	Z13 ζ	0.05581	≥ 0.5000	0.5025
190	Z13 ζ	0.04942	≥ 0.5000	0.5076
190	Z13 ζ	0.11160	≥ 0.5000	0.5136
190	Z13 ζ	0.07554	≥ 0.5000	0.5017

190	P13ξ	0.03661	≥ 0.5000	0.5929
190	P13ξ	0.03184	≥ 0.5000	0.5950
190	P13ξ	0.04777	≥ 0.5000	0.7908
190	P13ξ	0.07968	≥ 0.5000	0.5760
190	Z24ξ	0.05566	≥ 0.5000	0.5049
190	Z24ξ	0.06355	≥ 0.5000	0.5015
190	Z24ξ	0.12550	≥ 0.5000	0.9193
190	Z24ξ	0.07521	≥ 0.5000	0.5031
190	P24ξ	0.03661	≥ 0.5000	0.5016
190	P24ξ	0.03661	≥ 0.5000	0.5016
190	P24ξ	0.04786	≥ 0.5000	0.5026
190	P24ξ	0.08003	≥ 0.5000	0.5210
190	DC11	0.00000	≥ 0.9000	0.9444
190	DC12	0.00000	≥ 0.9000	0.9916
190	DC13	0.00000	≥ 0.9000	0.9916
190	DC24	0.00000	≥ 0.9000	0.9046

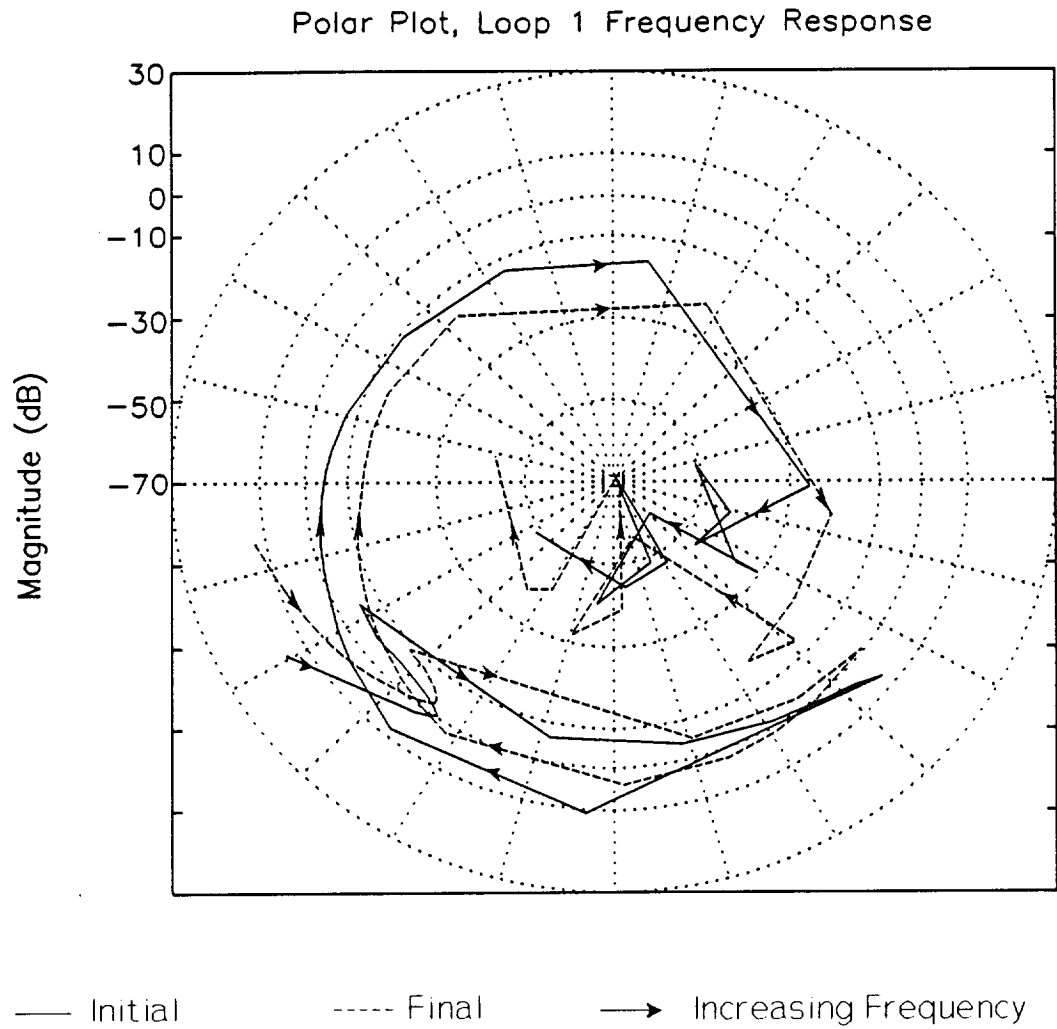


Figure 3.64: Polar Plot of the Compensated Open Loop System from control input $u(1)$ to measured output $y(1)$

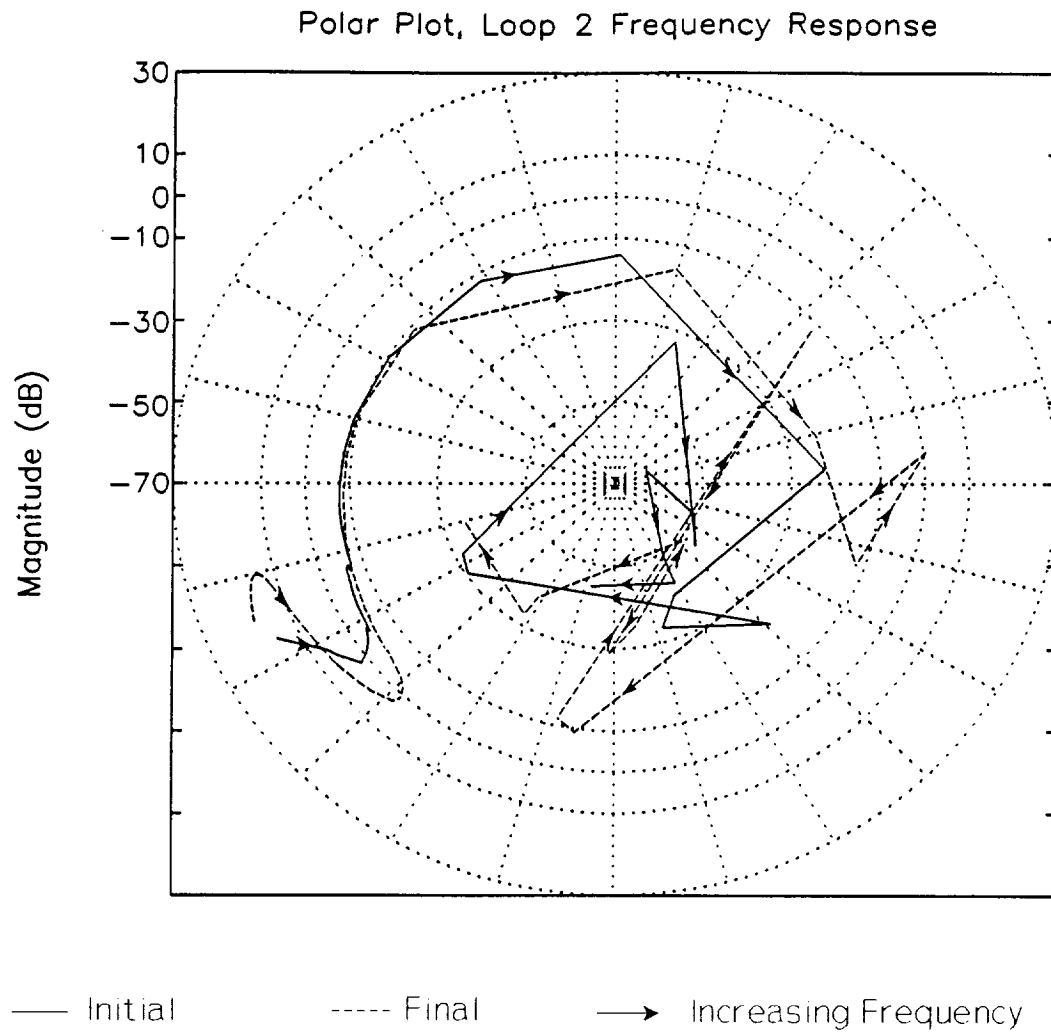


Figure 3.65: Polar Plot of the Compensated Open Loop System from control input $u(2)$ to measured output $y(2)$.

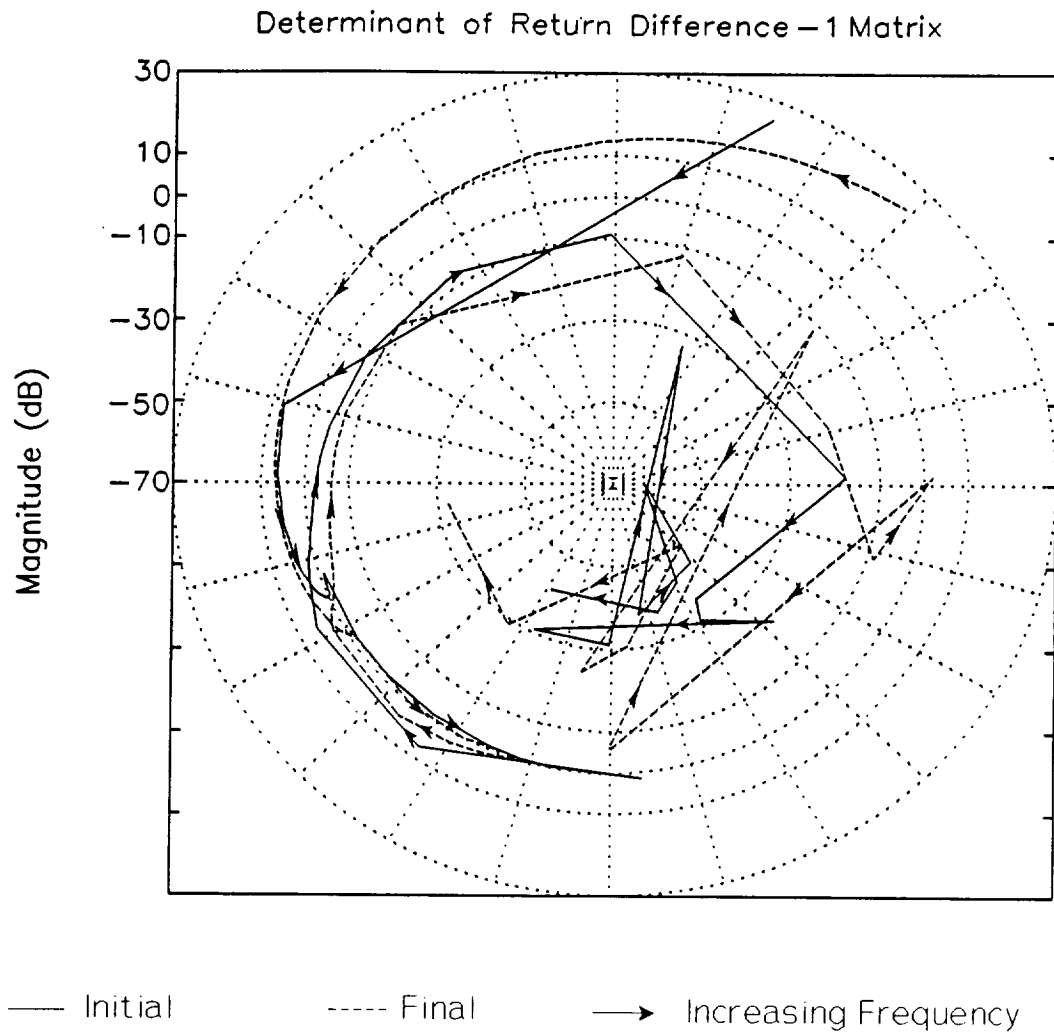


Figure 3.66: Polar Plot of the Frequency Response of the Determinant of the Return Difference - 1 Matrix.

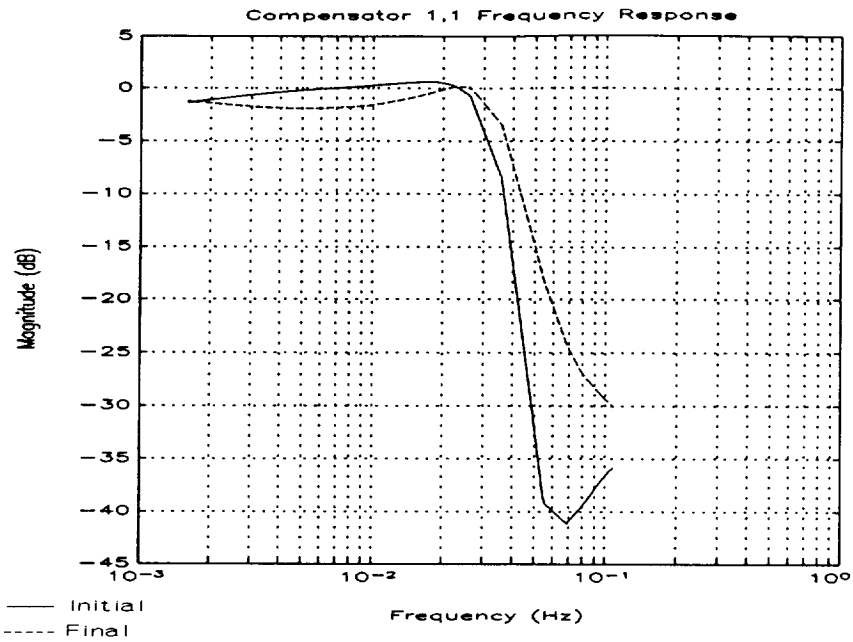


Figure 3.67: Magnitude Frequency Response of Compensator 1,1.

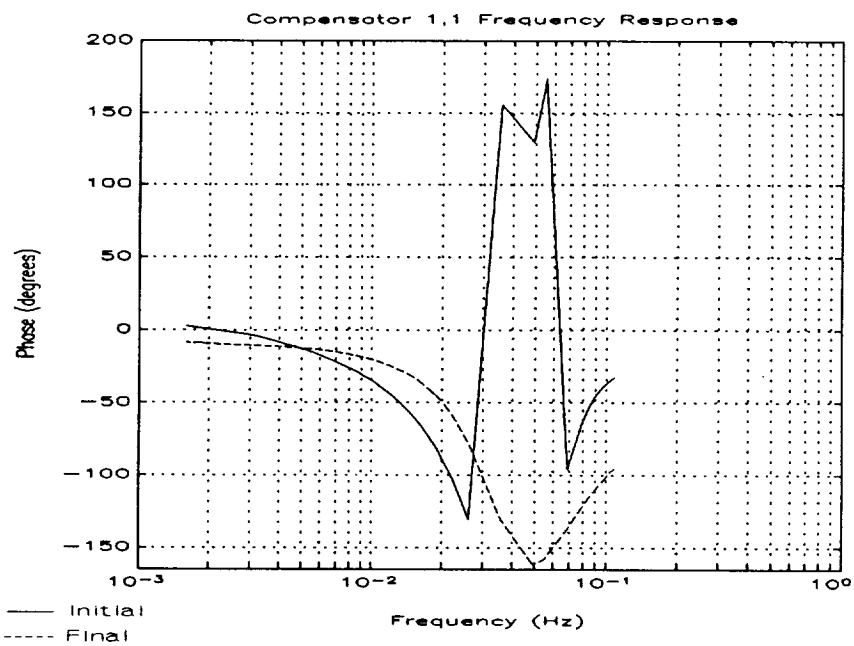


Figure 3.68: Phase Frequency Response of Compensator 1,1.

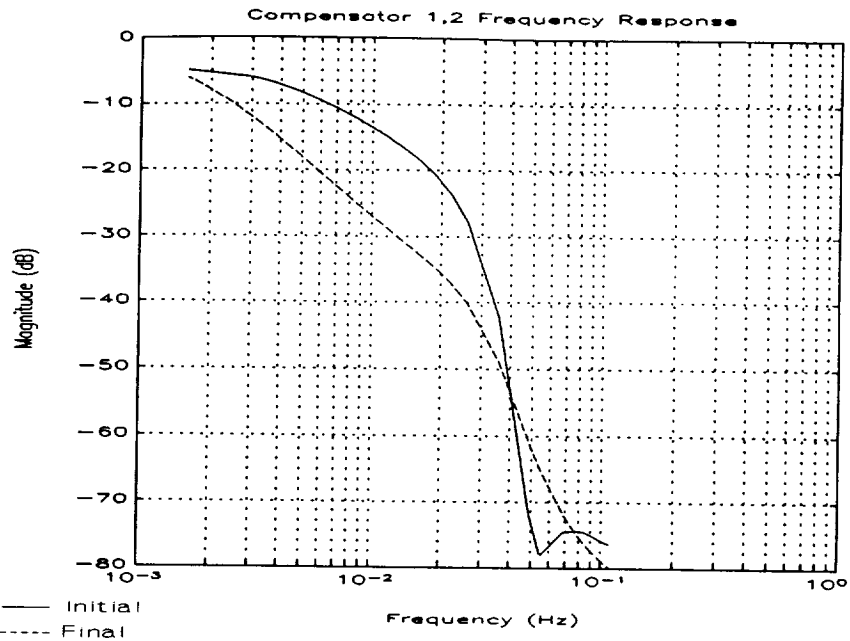


Figure 3.69: Magnitude Frequency Response of Compensator 1,2

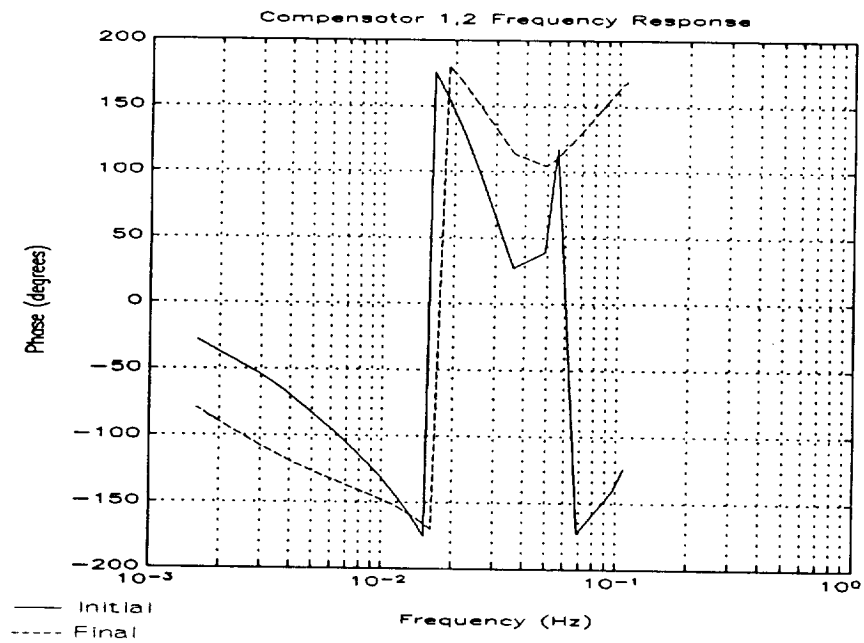


Figure 3.70: Phase Frequency Response of Compensator 1,2.

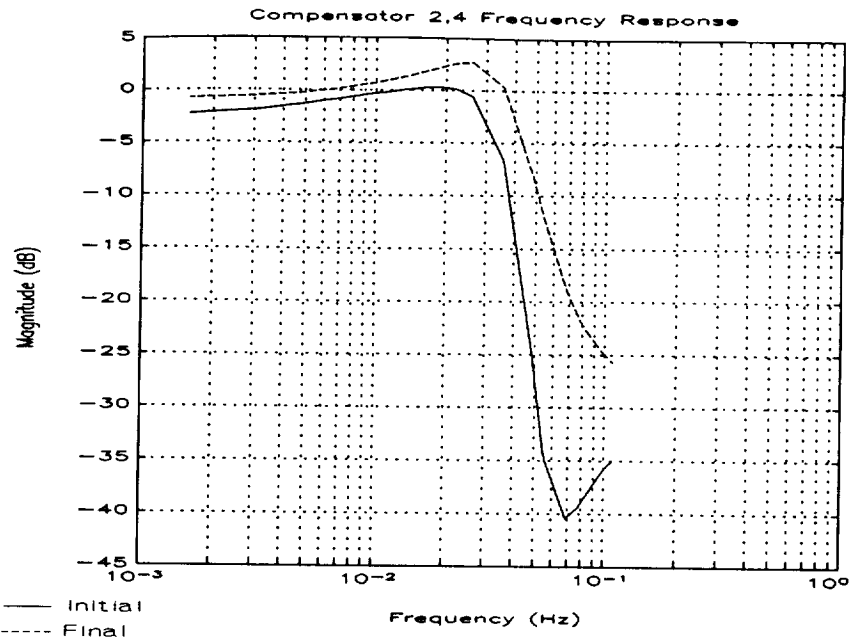


Figure 3.73: Magnitude Frequency Response of Compensator 2,4.

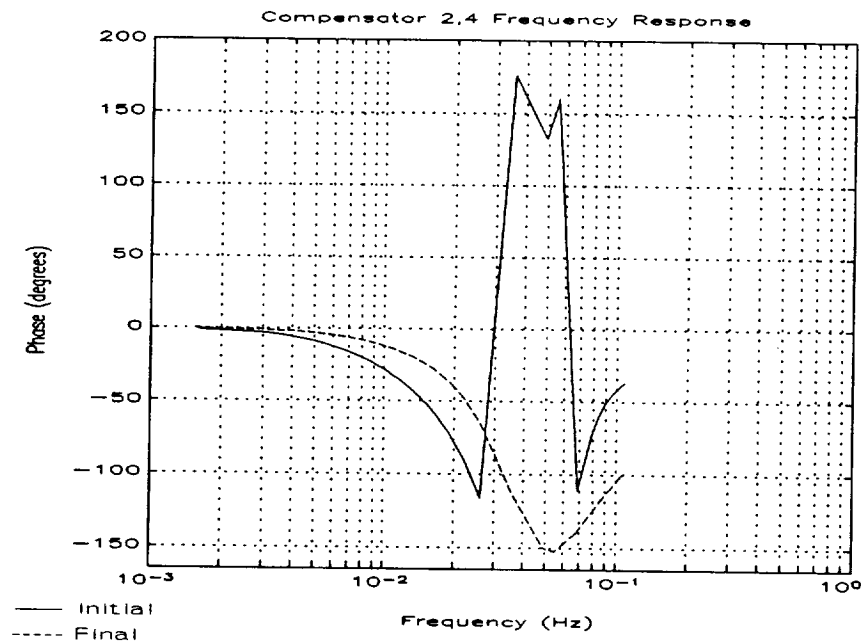


Figure 3.74: Phase Frequency Response of Compensator 2,4.

3.8 Conclusions and Recommendations for CIP

CIP is a numerical algorithm-based program designed for iterative improvement of MIMO control systems. Initially developed in the 1970's for use in improving open-loop frequency response specifications, it has undergone many changes over the years. The current version, OUCIP, contains enhancements including graphical user interface (GUI), closed-loop specifications, and an improved directional vector routine.

The work presented in this report includes theory and methods used in enhancements implemented in OUCIP. It was decided that the best method of testing stability of MIMO systems was to implement the generalized Nyquist criterion on the determinant of the return difference matrix, $\det[I + G(s)]$. The original CIP used a loop-at-a-time approach that proved to be unreliable. Each example shows significant improvement in the final design in terms of compensator damping ratios and DC gains, as well as relative stability design specifications and disturbance rejection characteristics (when specified).

Although CIP has reached a significant level of maturity, there are other enhancements that will add value to the code. One area that needs improvement is the entry and manipulation of data. Currently the user must create data files for each design improvement effort. Eventually OUCIP should have windows in which the user can key in the data which will be automatically stored by OUCIP into the proper files. This should be transparent to the user. In addition, the user should have the choice of entering the compensation matrix as transfer functions or state space realizations. Currently, the compensation matrix must be given in the form of transfer functions comprised of cascaded first and second order polynomials.

4 Model and Data Oriented Computer Aided Controller Design System

This section contains an in-depth discussion of the Model and Data Oriented Computer Aided Controller Design System. As stated earlier, this controller design system is based on a search technique that systematically alters the free parameters of an existing nominal controller to achieve multiple design constraints.

The initial discussion in sub-section 4.1 reviews some common design specifications used for linear, time-invariant, multi-input/multi-output systems. Since a computer aided approach is being considered, a brief discussion is then included in sub-section 4.2 that covers the general principles of mathematical programming.

Sub-section 4.3 begins with an examination of two simple previously existing algorithms for solving general inequality constrained problems. Sub-section 4.3 then continues with a discussion of two, also previously existing, more sophisticated and reliable methods that lend themselves primarily to controller design problems. Specifically, sub-section 4.3.2 discusses the CIT algorithm which is the heart of CIP, and includes a brief discussion of its philosophy and modes of operation. Sub-section 4.3.3 examines the Polak-Mayne algorithm and comments on the differences between this algorithm and CIT. Three new algorithms are introduced in sub-section 4.4 and their respective strengths and weaknesses are examined.

The following sub-section, 4.5, discusses various controller parameterizations and what effects a particular method of representation might have on the characteristics of the parameter space.

Sub-section 4.6 then deals with the computational tools necessary for implementation of the various gradient search algorithms. Specifically, the techniques for computing the constraint functions and their partial derivatives are presented. These constraint functions include controller frequency response constraints, transfer function frequency response constraints, frequency dependent singular value constraints, pole and zero damping ratio constraints, eigenvalue constraints and others.

The discussion then continues in sub-section 4.7 by examining methods by which an initial controller can be obtained. Two methods are included: a classical graphical approach, and an analytical synthesis method.

Two successful applications of these methods are then presented in sub-section 4.8. The first is the Active Control Technique Evaluation for Spacecraft (ACES)

facility located at the NASA Marshall Space Flight Center. The second is the redesign of the pointing control system on the Hubble Space Telescope.

Sub-section 4.9 contains a comparison of the performance of two of the more robust algorithms. Conclusions are then drawn concerning the various techniques and further research areas are cited in sub-section 4.10

4.1 Control System Design Specifications

This sub-section reviews some common LTI MIMO control system design specifications. These specifications are motivated by appealing to a typical design problem. The block diagram shown in Figure 4.1 illustrates a simplified model of a feedback control system. The variables in the block diagram are defined as follows.

Transfer function matrix blocks:

K : Controller
 G_p : Plant
 G_d : Disturbance model

Signals:

u : Control
 d : Disturbance
 y : Output to be controlled

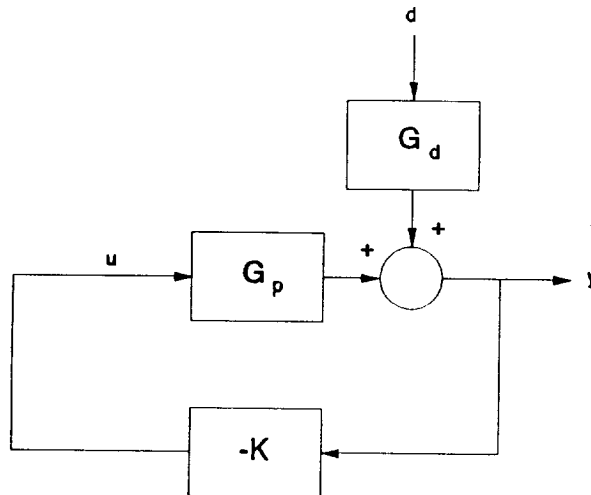


Figure 4.1 Block Diagram for Illustrating Design Constraints

It can be shown that the following relationships hold:

$$y = (I + G_p K)^{-1} G_d d \quad (4.1)$$

and

$$u = -(I + K G_p)^{-1} K G_d d . \quad (4.2)$$

In general, it is desired to design the controller such that (1) the closed-loop system has sufficient stability robustness to uncertainties and variations in the plant, (2) the effect of disturbances on the output is minimal, and (3) the control signal does not drive the plant's actuators into saturation.

Although many models of plant uncertainty can be devised, two of the most commonly used are in terms of multiplicative plant uncertainty modeled by $(I + \Delta_{mo})G_p$ (output uncertainty) and $G_p(I + \Delta_{mi})$ (input uncertainty). When these uncertainties, Δ_{mo} and Δ_{mi} , are not specified explicitly, but are given only in terms of their size (norm), they are often described as being *unstructured*. It can be shown (Maciejowski, 1989, p. 115) that in order to maintain closed-loop stability in the face of these unstructured uncertainties, it is sufficient to satisfy the following conditions for every $\omega \in R$:

$$\sigma_{\max} \left[G_p(j\omega) K(j\omega) (I + G_p(j\omega) K(j\omega))^{-1} \right] < 1 / \sigma_{\max}(\Delta_{mo}(j\omega)) , \quad (4.3)$$

and

$$\sigma_{\max} \left[K(j\omega) G_p(j\omega) (I + K(j\omega) G_p(j\omega))^{-1} \right] < 1 / \sigma_{\max}(\Delta_{mi}(j\omega)) , \quad (4.4)$$

where the symbol $\sigma_{\max}(\cdot)$ is the maximum singular value of a matrix. It is also assumed that the number of unstable poles of the plant does not change.

The use of singular values and the singular value decomposition (SVD) in control system analysis and design has become widespread in recent years. Many properties of the SVD are described in sub-section 4.6, but one of the most important is the following relationship between the maximum singular value and the matrix 2-norm, i.e.,

$$\sigma_{\max}(A) = \max_{\|x\|_2=1} \|Ax\|_2 , \quad (4.5)$$

where A is a matrix and x is a vector of compatible dimension. When the matrix A is the frequency response of a transfer function matrix evaluated at a single frequency, the maximum singular value can be thought of as the maximum possible gain (in the 2-norm sense) of that transfer function matrix at that frequency. Therefore the

stability robustness constraints described by Equations 4.3 and 4.4 are actually bounds on the gains of particular transfer function matrices.

In some situations it is important to provide stability robustness with respect to structured uncertainties, such as uncertainties in the frequencies and damping ratios of poles and zeros of the plant. Some stability robustness to these uncertainties can be achieved by placing lower bounds on the damping ratios of the controller's poles and zeros, thus preventing the controller from using lightly damped poles and zeros to cancel the plant's lightly damped dynamics. This cancelling phenomenon, sometimes called *plant inversion*, is a common problem with some modern analytical design techniques¹.

In Equations 4.3-4.4 the constraints are specified at each frequency explicitly. Sometimes, as in the case of H_∞ control theory in particular, the constraints are specified on a more global measure of a transfer function matrix, e.g.,

$$\|(I + G_p K)^{-1} G_d\|_\infty \leq c_1 \quad (4.6)$$

and

$$\|(I + K G_p)^{-1} K G_d\|_\infty \leq c_2 , \quad (4.7)$$

where c_1 and c_2 are nonnegative constants, and $\|\cdot\|_\infty$ is the operator infinity norm, which is defined for transfer function matrices as

$$\|T\|_\infty \triangleq \sup_{\omega} \sigma_{\max}(T(j\omega)) . \quad (4.8)$$

To give further meaning to $\|T\|_\infty$, suppose that the operator (transfer function) T is stable and proper (realizable), that the signal d is square integrable (has finite energy) as defined by

$$\|d\|_2 \triangleq \left[\int_{-\infty}^{\infty} d^T(t) d(t) dt \right]^{1/2} < \infty , \quad (4.9)$$

and that

$$y = Td , \quad (4.10)$$

i.e., T operates on d to yield y . Then it can be shown (Francis, 1987) that

¹ Some recent workers have proposed plant inversion as a design technique itself. The authors believe the robustness problems of such an approach make plant inversion unsuitable for structural control.

$$\|T\|_{\infty} = \sup_d \frac{\|y\|_2}{\|d\|_2} . \quad (4.11)$$

Thus the constraint specified by Equation 4.6 can be interpreted as an attempt to maintain good closed-loop performance by limiting the impact that a worst case disturbance has upon the output. Likewise, Equation 4.7 can be interpreted as an attempt to avoid actuator saturation by limiting the impact that a possibly different worst case disturbance has upon the control signal. Equation 4.6 reveals that K needs to be large in order to achieve good performance, but Equation 4.7 reveals that making K large increases the magnitude of the control signals, which can be an undesirable effect. This brings to light one of the strongest motivations for continuing the development of search-based methods for control system design: they have the ability to simultaneously handle such conflicting design tradeoffs as individual hard constraints rather than as a single cumulative constraint.

Another type of performance constraint is a bound on an average gain of a transfer function. A common constraint of this type is to specify a limit for the root-mean-square (RMS) value of an output signal that results from a stochastic input signal possessing an identity covariance matrix. To be more precise, it is desired to place an upper bound on $E\{y^T y\}$, where $y = Td$, d is a stochastic signal with an identity covariance matrix, and E is the expectation operator. This constraint can be expressed in terms of the frequency response of a transfer function matrix as

$$\|(I + G_p K)^{-1} G_d\|_2 \leq c_3 , \quad (4.12)$$

where c_3 is a positive constant, and the operator 2-norm has been used (Postlethwaite *et al.*, 1981). This norm is defined by

$$\|T\|_2 \triangleq \left\{ \frac{1}{\pi} \int_0^{\infty} \text{tr}[T^H(j\omega) T(j\omega)] d\omega \right\}^{1/2} , \quad (4.13)$$

where T is a transfer function matrix. In this sense, Equation 4.12 places an upper bound on the RMS value of that component of the output signal that is due to the disturbance signal.

The aspects of control systems design presented in this sub-section have been rather limited in scope. Other specifications on such items as interaction of particular outputs and inputs, noise suppression, command tracking errors, and steady-state

errors, to name a few, can also be formulated as constraints on various functions of the plant and controller.

4.2 Mathematical Programming Principles

As was illustrated in sub-section 4.1, many control system design constraints can be stated in terms of precise but usually nonlinear inequality constraints that are functions of the controller parameters. In the terminology of mathematical programming, this type of problem is usually referred to as an inequality constraint problem (IP). Standard algorithms for solving IP problems are almost always designed to optimize some functional, subject to a set of inequality constraints, and they require that the initial parameter vector be feasible, i.e., the inequalities must be initially satisfied. Examples of such techniques are active set methods and barrier function methods (Luenberger, 1984, Chaps. 11 and 12), to name only two. These methods are not directly applicable to search-based methods for controller design in which the primary goal is to find a point in the feasible region (the set of all controllers of a fixed order that satisfy all the design constraints); not optimization of a functional. Another aspect of the controller design problem that distinguishes it from the standard IP problem is that the number of inequality constraints is almost always much greater than the number of free parameters, thus eliminating the possibility of using a nonlinear equation solver to try to find a point in the feasible region. Therefore, special techniques must be applied in order to develop effective algorithms for this type of IP problem.

The majority of mathematical programming algorithms, including those that have been developed specifically for search-based controller design, possess the following model algorithm structure.

Standard Model Algorithm

Let $x \in R^N$ be the current parameter vector.

- Step 1. [Test for a solution.] If the conditions for a solution are satisfied, the algorithm terminates with x as the solution.
- Step 2. [Compute a search direction.] Compute a non-zero $d \in R^N$; the direction of search.

- Step 3. [Compute a step length.] Compute a positive step $\alpha \in \mathbb{R}$ along d for which a measure of the algorithm's progress is improved.
- Step 4. [Update the parameter vector.] Set $x := x + \alpha d$. Go to step 1.

It is well established that step 2, computation of the search direction, is the crucial step in determining the performance of an algorithm with this structure, although some multi-step algorithms, i.e., algorithms in which the search direction at the current iteration depends explicitly on previous search directions, such as the conjugate gradient method, are highly sensitive to the choice of step length (Luenberger, 1984, p. 257).

Probably the most important mathematical principle used in determining the search direction is the following result that is based upon the Taylor series and proved in Appendix A.

Claim 1.

Suppose $f: \mathbb{R}^N \rightarrow \mathbb{R}$ has the second-order Taylor series expansion

$$f(x + hd) = f(x) + hDf(x)d + \frac{1}{2}h^2d^TD^2(x + h\theta d)d, \quad (4.14)$$

where $x \in \mathbb{R}^N$, $Df(x) \in \mathbb{R}^{1 \times N}$ is the gradient of f evaluated at x as defined by

$$Df(x) \triangleq \left[\frac{\partial f}{\partial x_1}(x) \quad \frac{\partial f}{\partial x_2}(x) \quad \dots \quad \frac{\partial f}{\partial x_N}(x) \right], \quad (4.15)$$

and $D^2f(x) \in \mathbb{R}^{N \times N}$, often referred to as the Hessian of f evaluated at x , is a matrix with its (i, j) element defined as

$$[D^2f(x)]_{ij} = \frac{\partial^2 f}{\partial x_i \partial x_j}(x) \quad (4.16).$$

Also suppose that $\|Df(x)\| \neq 0$.

Then there exist d and h such that $f(x + hd) > f(x)$.

This result reveals that there exists a "direction" d and a "distance" h for which x can be changed such that the value of the function f is increased. The significance of this

result to mathematical programming algorithms is immediately apparent: in order to improve a violated constraint (e.g., increase its value), it is sufficient to find a search direction and step length that satisfy the hypotheses of the above claim. In problems of greater than one dimension the number of these feasible search directions is uncountable, and the algorithm must decide which of these directions is the "best".

In the case of unconstrained minimization of a single objective function, the algorithm that uses the negative of the gradient as the search direction is the well-known *method of steepest descent*. Although the linear convergence rate of this method leaves something to be desired in the case of optimization problems, its global convergence properties (i.e., it will not diverge) make it very useful when the current parameter vector is far from the actual solution. When near the optimal solution, an algorithm with a better convergence rate, such as Newton's method, a quasi-Newton method, or a conjugate gradient method, is often used (Luenberger, 1984, chaps. 7, 8, and 9).

In inequality constraint problems with no functional to optimize, the importance of the rate of convergence near the optimum completely loses its meaning and is replaced by the concept of rate of convergence toward the feasible region. How this can modify search direction calculations is illustrated by Figure 4.2, where the contours lines of a function of two variables are drawn. In order to take the shortest step to the feasible region (shaded area) from the point x_0 , a search direction along the direction of gradient g is actually superior to the direction d that takes a step toward the minimum of the function. This and other issues pertaining to purely inequality constrained problems are discussed further in sub-sections 4.3 and 4.4.

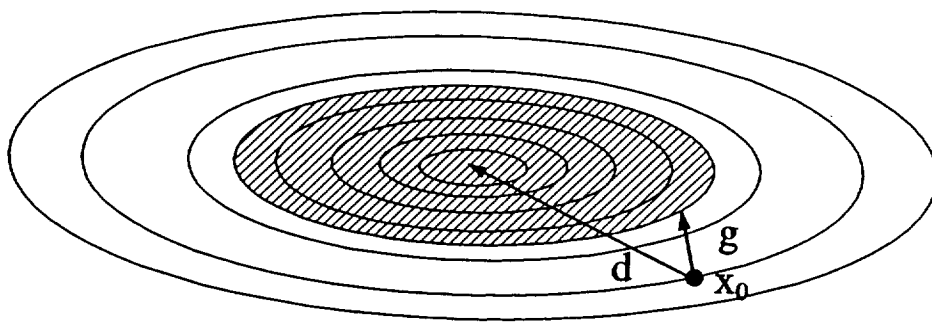


Figure 4.2 Illustration of Search Direction Calculations

4.3 Previously Developed Algorithms

In this sub-section two simple approaches to solving purely inequality constrained problems are presented, followed by detailed descriptions of two more sophisticated and reliable algorithms that have been developed primarily for controller design improvement.

4.3.1 Simple Approaches

Let

$$F = \{f_i(x) \leq 0, i = 1, 2, \dots, N_f\} \quad (4.17)$$

define a set of inequality constraints where $x \in R^N$, and $f_i: R^N \rightarrow R$, $i = 1, 2, \dots, N_f$, are continuously differentiable functions. Let

$$V(x) = \{i \mid f_i(x) > 0, i = 1, 2, \dots, N_f\} \quad (4.18)$$

denote the set of constraints that are violated. The question becomes: how should the parameter vector x be changed in order to bring the violated constraints closer to satisfaction? A tempting answer might be to try to convert the problem to an "optimization" problem by forming the objective function

$$m(x) = \frac{1}{2} \sum_{i \in V(x)} f_i^2(x) , \quad (4.19)$$

setting the search direction d equal to the negative of the gradient of this function, i.e.,

$$d = - \sum_{i \in V(x)} f_i(x) Df_i(x) , \quad (4.20)$$

and finding a step length α such that the condition

$$\sum_{i \in V(x + \alpha d)} f_i^2(x + \alpha d) < \sum_{i \in V(x)} f_i^2(x) \quad (4.21)$$

is satisfied. This is, in fact, the basis of a method that has been previously used for controller design improvement (Newsom and Mukhopadhyay, 1984). The disadvantage to this approach is that by including all the constraint violations in the objective function, the greatest violation can easily be dominated by the potentially numerous lesser violations and can actually become much worse even while the objective function improves. Experience also indicates that including all the constraint

violations in a single objective function tends to create local minima that cause the algorithm to *jam* (terminate) before all the constraints are satisfied.

At the other extreme, another approach that might be tempting would be to simply set the search direction equal to the negative of the gradient of the single worst violation, i.e., $d = -Df_{\text{worst}}(x)$ and find a step length such that the condition

$$f_{\text{worst}}(x + \alpha d) < f_{\text{worst}}(x) \quad (4.22)$$

is satisfied. However, this approach is highly subject to jamming. To understand why this is the case, consider the situation when the worst violation and the next to worst violation are nearly equal in value. Also suppose that the gradients of the functions associated with these two violations have the property that $Df_{\text{worst}}(x) Df_{\text{next}}(x)^T < 0$, thus implying that these constraints are, in a sense, conflicting. If the search direction d is set equal to $-Df_{\text{worst}}(x)$, then $-Df_{\text{next}}(x)d < 0$. This reveals that the constraint associated with $f_{\text{next}}(x)$ is likely to get worse. Since the two violations are nearly equal in value, it is quite possible that a step length within the precision of the computer cannot be found such that an improvement is achieved.

To overcome the disadvantages of these two simple approaches, it is necessary to more intelligently take into account the relationships that the constraints have with one another. Two algorithms developed for controller design improvement that do this are now presented.

4.3.2 The Constraint Improvement Technique

The first algorithm to be described, the Constraint Improvement Technique (CIT), was one of the first search-based algorithms developed for controller design improvement and has been successfully applied to controller design problems with phase margin, gain margin, stability margin (the closest approach of the compensated open-loop frequency response to the $-1 + j0$ point in the complex plane), attenuation margin (the distance from the compensated open-loop frequency response to the origin of the complex plane), and zero-frequency (d.c.) gain constraints (Mitchell, 1972). This algorithm follows the control flow of the Standard Model Algorithm presented in sub-section 4.2. The problem to be solved may be stated mathematically as

Find $\mathbf{x} \in R^N$ to satisfy

$$c_i(\mathbf{x}) \leq 0, \quad i = 1, 2, \dots, N_c,$$

$$f_i(\mathbf{x}) = \max \{ h_i(\omega_j, \mathbf{x}), \omega_j \in \Omega_i \} \leq 0, \quad i = 1, 2, \dots, N_f$$

where each Ω_i is finite collection of frequencies for which the i^{th} constraint is defined.

The constraints corresponding to the f_i are frequency dependent constraints such as gain margins, phase margins, stability margins, and attenuation margins. The quantity h_i defines a mathematical relationship between the i^{th} performance measure, the frequencies in Ω_i , and the controller parameters in \mathbf{x} . The constraints corresponding to the c_i are d.c. gain constraints and damping ratio constraints.

To describe CIT, first denote the frequency dependent constraint violations and ordinary constraint violations by the sets

$$V_f = \{(i, j) \mid h_i(\omega_j, \mathbf{x}) > 0\} \quad (4.23)$$

and

$$V_c = \{i \mid c_i(\mathbf{x}) > 0\} \quad (4.24)$$

respectively. The search direction \mathbf{d} is calculated by requiring that

$$-Dh_i(\omega_j, \mathbf{x})\mathbf{d} = 1, \quad \forall (i, j) \in V_f \quad (4.25)$$

and

$$-Dc_i(\mathbf{x})\mathbf{d} = 1, \quad \forall i \in V_c. \quad (4.26)$$

Taken together Equations 4.25 and 4.26 form a system of linear equations in the elements of \mathbf{d} that can be written in the form

$$J(\mathbf{x})\mathbf{d} = \mathbf{p}, \quad (4.27)$$

where the rows of $J(\mathbf{x})$ are the gradients of the violated constraints, and \mathbf{p} is a vector of 1's. A solution to these equations exists as long as \mathbf{p} is in the range of $J(\mathbf{x})$. This is usually the case since the number of gradients is not allowed to exceed the number of parameters N_p . In fact, the system is usually underdetermined, meaning that there are an infinite number of solutions. When this is the case, the solution having the smallest 2-norm is used. It is important to realize that satisfaction of these equations

guarantees that the dot product between the search direction and the negative of each gradient is positive, satisfying Claim 1 of sub-section 4.2. This means that the resulting search direction is a feasible direction for each constraint violation that is to be decreased. Empirical evidence indicates that when the gradients have norms that range over several orders of magnitude, better algorithm performance can be achieved if each of the gradients is normalized to unit length before the search direction is calculated. This might be explained by the fact that without gradient normalization the search direction generally places it greatest emphasis on the gradient with the smallest norm.

CIT allows the user to choose between two different measures of constraint improvement. One method is the Total Improved Frequency Response (TIFR); the other is the Sum Improved Frequency Response (SIFR). When operated in the TIFR mode the algorithm accepts the trial step length only if *all* the constraint violations for which the gradients were computed improve. When operated in the SIFR mode the algorithm accepts the trial step length if the *sum* of the constraint violations for which the gradients were computed improves. Although the TIFR mode has the apparent advantage that it requires all of the constraint violations to improve at each step, experience has shown that the SIFR mode tends to give better overall algorithm performance.

CIT has recently been modified to include a new search direction calculation defined by

$$\mathbf{d} = -\arg \max_{\mathbf{q}} \left[\sum_{(i,j) \in V_f} Dh_i(\omega_j, \mathbf{x}) \mathbf{q} + \sum_{i \in V_c} Dc_i(\mathbf{x}) \mathbf{q} \right] \quad (4.28)$$

subject to

$$Dh_i(\omega_j, \mathbf{x}) \mathbf{d} \geq 0, \quad \forall (i,j) \in V_f, \quad (4.29)$$

$$Dc_i(\mathbf{x}) \mathbf{d} \geq 0, \quad \forall i \in V_c, \quad (4.30)$$

and $\|\mathbf{d}\|_2 = 1$. This search direction, which attempts to maximize the sum of the vector inner products while preventing any single inner product from being negative, tends to place its greatest emphasis on the largest gradient. When using this search direction CIT is always operated in the SIFR mode. Preliminary studies using this search direction indicate improvements in algorithm performance over using the original search direction.

A key element of the implemented version of CIT, called the Compensator Improvement Program (CIP), is a special technique that is employed to avoid the potential difficulties that are associated with the discontinuous behavior of gain margin and phase margin constraints (the frequencies at which they occur can change as the controller frequency response changes). This special technique, described in detail by Mitchell (1972), is based upon the idea of "pushing" or "pulling" the frequency response near the frequency at which a margin occurs in such a way as to guarantee that the violated margin cannot become worse even if the frequency at which the margin occurs changes as the controller is changed. Discontinuities associated with stability margin and attenuation margin constraints are overcome by including in the search direction calculation the gradients of those constraint functions that correspond to "peaks" in the measures of those margins.

The algorithm is terminated either when all the constraints are satisfied, the norm of a gradient of a violated constraint becomes sufficiently small, indicating that no further local improvement in that constraint is possible, or a step length cannot be found that is greater than a user specified minimum value. This second condition is often caused by constraints that are not locally consistent.

Although experimental evidence indicates that CIT is more efficient when operated in SIFR mode than TIFR mode, the possibility exists that the algorithm will zigzag as highly sensitive constraints move from violation to satisfaction and back into violation again. It may be possible to overcome this difficulty by introducing information from previous iterations into the search direction calculation. Another potential difficulty with the SIFR mode, or any algorithm that uses a cumulative measure, is that it is possible that the worst constraint violation will become worse instead of better. This can cause difficulties if the constraint that is associated with the worst violation is a stability margin, since the algorithm may cause the closed-loop system to become unstable. The following algorithm uses the worst constraint violation as the measure of algorithm progress, yet it does not require all constraint violations to improve. Thus it is, in a sense, a mix of the TIFR and SIFR modes.

4.3.3 The Polak-Mayne Algorithm

An algorithm that uses a somewhat different approach than CIT, but still follows the control flow of the Standard Model Algorithm has been proposed by Polak and Mayne (1976). The controller design problem to be solved may be stated mathematically as

Find $\mathbf{x} \in R^N$ to satisfy

$$c_i(\mathbf{x}) \leq 0, \quad i = 1, 2, \dots, N_c,$$

$$f_i(\mathbf{x}) = \max\{h_i(\omega, \mathbf{x}), \omega_j \in \Omega_i\} \leq 0, \quad i = 1, 2, \dots, N_f$$

where each $\Omega_i \subset R$ is a compact set on which the i^{th} constraint is defined.

The functional constraints, the f_i , can be used to specify the "shapes" of various frequency responses. Since it is very impractical to accurately evaluate each f_i at every iteration, each Ω_i is adaptively discretized at each iteration to contain a finite number of frequencies. Of course, if a design is performed using a nonparametric plant model, then the number of frequency points is already finite. After discretization the problem statement becomes

Find $\mathbf{x} \in R^N$ to satisfy

$$c_i(\mathbf{x}) \leq 0, \quad i = 1, 2, \dots, N_c,$$

$$f_i(\mathbf{x}) = \max\{h_i(\omega_j, \mathbf{x}), \omega_j \in \Omega_i\} \leq 0, \quad i = 1, 2, \dots, N_f$$

where each Ω_i is a finite collection of frequencies for which the i^{th} constraint is defined.

At each iteration the search direction is calculated by finding the \mathbf{d} that solves

$$\theta = \min_{\mathbf{d}} \max \{Dc_i(\mathbf{x})\mathbf{d}, i \in V_c; Dh_i(\omega_j, \mathbf{x})\mathbf{d}, (i, j) \in V_f\}, \quad (4.31)$$

where $\|\mathbf{d}\|_{\infty} \leq 1$, and the sets V_f and V_c are determined by defining ϵ -active constraints as described below. Let

$$\psi(\mathbf{x}) = \max\{c_i(\mathbf{x}), i = 1, 2, \dots, N_c; f_i(\mathbf{x}), i = 1, 2, \dots, N_f\} \quad (4.32)$$

denote the value of the worst violation. Then for $\epsilon > 0$ define

$$V_c = \{i | c_i(\mathbf{x}) > \psi(\mathbf{x}) - \epsilon\} \quad (4.33)$$

to be the set of ϵ -active ordinary constraints and

$$V_f = \{(i,j) | h_i(\omega_j, x) > \psi(x) - \epsilon, \omega_j \in \Omega_i\} \quad (4.34)$$

to be the set of ϵ -active frequency dependent constraints. The purpose in defining ϵ -active constraints is to force the algorithm to try to always improve the worst constraint violation, while trying to avoid the type of jamming discussed in sub-section 4.3.1. Notice that this method of calculating the search direction differs somewhat from the method used in CIT in that the ϵ -active constraint technique compares all constraint violations (ordinary and frequency dependent) to the worst constraint violation when deciding which gradients to use for the search direction calculation, whereas in CIT every frequency dependent constraint that has a violation contributes at least one gradient.

In an attempt to obtain a search direction that has a sufficiently negative dot product with each of the gradients to be considered, the solution to Equation 4.31 is required to satisfy $\theta < -2\epsilon$. If it does not, the nominal value of ϵ is reduced (in effect, reducing the number of gradients in the search direction calculation), and the search direction recalculated until this requirement is met.

The step length α is determined by requiring that

$$\psi(x + \alpha d) < \psi(x) - \alpha\beta\epsilon, \quad (4.35)$$

where $\beta \in [0,1)$ is a user specified parameter that helps to guarantee a "sufficient improvement". The algorithm is terminated whenever all the constraints are satisfied or if step length cannot be found that gives a sufficient improvement.

Although the search direction defined by Equation 4.32 has the property that it is a descent direction for the constraints within the ϵ -boundary, it generally places its greatest emphasis on the constraint gradient possessing the smallest norm without any consideration of the relative degree to which each of the constraints is violated or the behavior of the constraint violations outside the ϵ -boundary. Empirical evidence indicates that this often causes algorithm termination when the gradient corresponding to the worst constraint violation has a norm much larger than the other gradients. This is because the search direction can be almost orthogonal to the gradient of the worst violation and still have a dot product with that gradient that is less than -2ϵ . The algorithm also tends to make very slow progress when a constraint function possessing a large gradient moves inside and outside of the ϵ -boundary from iteration to iteration. These difficulties can be attributed to the fact that the search direction calculation does not actually correspond to the measure of constraint improvement.

Notice that the search direction, as calculated by Equation 4.32, does not place any special emphasis on the worst violation, even though this is the measure of constraint improvement.

4.4 Three New Algorithms

In this chapter, three new algorithms for search-based controller design improvement are presented. Each of the algorithms is similar in some respects to the algorithms presented in sub-section 4.3, but the methods used for determining the search direction, the step length, and the measure of constraint improvement are different in each case. The algorithms are presented in the chronological order in which they were developed and tested. The third algorithm, although not the most complex, has generally given the best performance and has been the most robust in the sense that occurrences of jamming are far less frequent than for the other algorithms. The mathematical problem statement for each of these algorithms is the same and is given by

Find $\mathbf{x} \in R^N$ to satisfy

$$c_i(\mathbf{x}) \leq 0, \quad i = 1, 2, \dots, N_c,$$

$$f_i(\mathbf{x}) = \max\{h_i(\omega_j, \mathbf{x}), \omega_j \in \Omega_i\} \leq 0, \quad i = 1, 2, \dots, N_f$$

where each Ω_i is finite collection of frequencies for which the i^{th} constraint is defined.

4.4.1 Algorithm A1

This algorithm follows the control flow of the Standard Model Algorithm and is the simplest of the algorithms presented in this sub-section. To develop the method used for determining the search direction, define the constraint violations to be considered at each iteration by

$$V_c = \{i | c_i(\mathbf{x}) > 0\} \tag{4.36}$$

and

$$V_f = \{(i, j) | h_i(\omega_j, \mathbf{x}) \text{ is among the } N_{\max} \text{ largest values for } i\}, \tag{4.37}$$

where $N_{max} > 0$ is a user specified integer. In other words, V_f is a set corresponding to the N_{max} worst violations for each frequency dependent constraint. The set V_f is defined in this manner for two reasons. The first reason is to ensure that each violated constraint contribute at least one gradient to the search direction calculation so that all the constraints have some chance of improving. The second reason is to try to prevent jamming by including several gradients that correspond to violations that are "near" in value to the worst violation for each frequency dependent constraint. This is similar to the ϵ -active constraint approach. The search direction d is calculated by requiring that

$$\frac{-Dc_i(x)d}{\|Dc_i(x)\|_2} = 1, \quad \forall i \in V_c \quad (4.38)$$

and

$$\frac{-Dh_i(\omega_j, x)d}{\|Dh_i(\omega_j, x)\|_2} = 1, \quad \forall (i, j) \in V_f. \quad (4.39)$$

Assuming that the system of linear equations corresponding to Equations 4.38 and 4.39 is consistent, it can be shown that the angle between the resulting search direction and each of the gradients is the same. Furthermore, it can also be shown that this angle satisfies the relationship

$$\cos(\theta) = 1/\|d\|_2, \quad (4.40)$$

where θ is the angle between the search direction and the negative of each of the gradients. Therefore, if the minimum 2-norm solution to Equations 4.38 and 4.39 is used, then θ is minimized. This is important since it is desirable to keep the search direction as close (in direction) as possible to the negative of the gradients since no further information about the behavior of the constraint functions is available.

The step length α is chosen such that all the violated, ordinary constraints and the worst violation of each frequency dependent constraint decrease. If a satisfactory step length cannot be found at some iteration before all the constraints have been satisfied, then one of two conditions usually exists. First, the norm of one or more of the gradients could be small, implying that local improvement in that constraint is difficult. Second, two or more gradients might be conflicting, which means that they are pointing in nearly opposite directions. Since Equation 4.40 reveals that the angle between the search direction and each of the gradients is equal, the search direction is virtually orthogonal to all the gradients in that case. This makes achieving an improvement in all the violated constraints extremely difficult. Sometimes this

difficulty can be avoided by noting that if one of the opposing gradients does not correspond to the worst violation of its corresponding frequency dependent constraint, then that gradient can be dropped from the search direction calculation without eliminating the possibility of decreasing all the violated constraints. If, however, the opposing gradients correspond to the worst violations of two constraints, then the two constraints are locally inconsistent. At this point the algorithm can either be terminated, or one of the constraints relaxed and the algorithm continued. A simplified flowchart of the algorithm is shown in Figure 4.3.

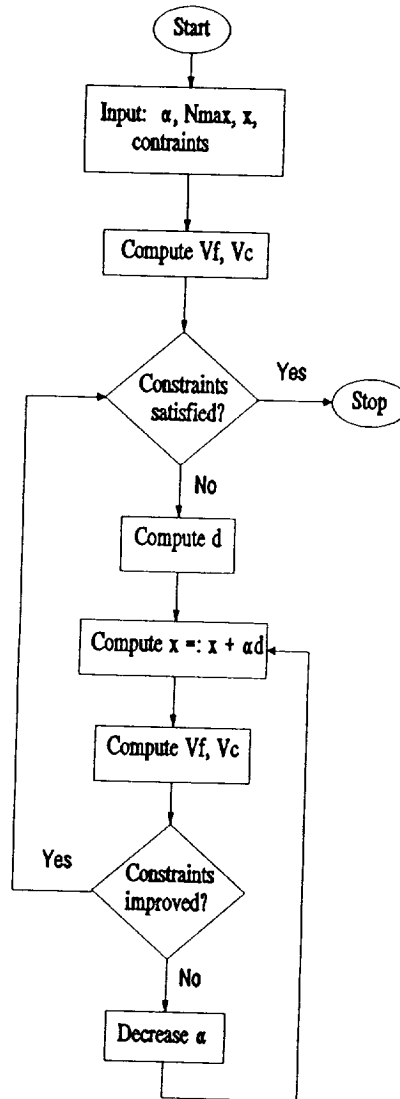


Figure 4.3 Simplified Flowchart of Algorithm A1

The application of this algorithm to controller design for a large space structure ground test facility is presented in sub-section 4.8. Although the resulting controller was successful upon implementation, a major weakness in the algorithm was discovered as described in sub-section 4.8. In an attempt to overcome this weakness, an algorithm closely resembling the Polak-Mayne Algorithm was developed and tested.

4.4.2 Algorithm A2

This algorithm is the most complex of the algorithms presented in this chapter primarily because of the introduction of special vectors other than gradients into the search direction calculation. The motivation for including more than gradient information in the search direction calculation is to try to overcome the fact that gradients are often poor predictors of function behavior when higher order terms in the Taylor series expansion are not negligible.

One approach to obtaining more information is to directly compute higher order partial derivatives. Even though the use of second-order information can significantly improve the performance of optimization algorithms, the calculation of these partial derivatives can be computationally expensive and analytically difficult. Another approach is to try to decompose a function into two functions such that one of the functions has predictable behavior. For example, suppose the real-valued function $f:R^n \rightarrow R$ can be decomposed into the two continuously differentiable functions $g:R^n \rightarrow R^p$ and $h:R^p \rightarrow R$ such that

$$f(x) = h[g(x)] \quad (4.41)$$

and such that h is a linear function of g , i.e.,

$$h(g(x + \delta x)) = h(g(x)) + \alpha \delta g \quad (4.42)$$

for some constant row vector α , where $\delta g = g(x + \delta x) - g(x)$ and $x \in R^n$. Therefore, it is possible to calculate a precise change in g given a desired change in h . Writing the first order Taylor series approximation of the vector-valued function g about x ,

$$g(x + \delta x) \approx g(x) + Dg(x)\delta x, \quad (4.43)$$

and solving for δx yields the desired parameter correction or *in-the-large* direction. As an example illustrating the calculation of an in-the-large direction, consider the

following property of the SVD of a complex-valued matrix. If $U\Sigma V^H = T$ is the SVD of $T \in \mathbb{C}^{m \times n}$ then it can be shown that the *singular value expansion* of T is given by

$$T = \sum_{i=1}^r \sigma_i u_i v_i^H, \quad r = \min\{n, m\}, \quad (4.44)$$

where the σ_i , u_i , and v_i are the singular values, left singular vectors, and right singular vectors of T , respectively. Now suppose it is desired to change the value of the maximum singular value σ_1 by changing the matrix T . Consider a change in T defined by $\delta T = (\delta\sigma_1)u_1 v_1^H$. If

$$\bar{T} = T + \delta T \quad (4.45)$$

then it is clear that

$$\bar{T} = \bar{\sigma}_1 u_1 v_1^H + \sum_{i=2}^r \sigma_i u_i v_i^H, \quad (4.46)$$

where $\bar{\sigma}_1 = \sigma_1 + \delta\sigma_1$. Thus, it is possible to determine a precise change in T that results in a predetermined change in σ_1 . Then if T is a function (possibly nonlinear) of a vector of parameters p , an approximation to the desired change in p can be calculated by solving the system of linear equations generated from a first-order approximation of T with respect to p , i.e.,

$$\frac{\partial T_{ij}}{\partial p}(p) \delta p = (\delta T)_{ij}, \quad i = 1, 2, \dots, m, \quad j = 1, 2, \dots, n. \quad (4.47)$$

The reason that the in-the-large direction δp contains information about the behavior of σ_1 as a function of p in addition to the gradient is clear: the fact that σ_1 is a linear function of the elements of T has been explicitly taken into account.

Now to describe the complete algorithm, including the in-the-large directions, define the set of ϵ -active frequency dependent constraints by

$$V_f = \{(i, j) | h_i(\omega_j, x) > f_i(x) - \epsilon, \quad \omega_j \in \Omega_i\} \quad (4.48)$$

where $\epsilon > 0$. Define the violations of the ordinary constraints by

$$V_c = \{i | c_i(x) > 0\} \quad (4.49)$$

The set V_f is similar to the set of ϵ -active constraints defined in the Polak-Mayne Algorithm except that here the ϵ -active constraints are defined for *each* frequency dependent constraint, and they do not involve the ordinary constraints. The reason for these differences is to force every violated constraint to contribute at least one

gradient to the search direction calculation so that all the violated constraint have the possibility of being improved at each iteration.

When using the in-the-large directions in the search direction calculation it is not sufficient to simply require that the search direction have a positive inner product with each of the in-the-large directions to guarantee that all the violated constraints have the possibility of decreasing. This is because the resulting search direction may have a positive inner product with a gradient of one or more of the violated constraint functions that needs to be decreased. Therefore, it is also necessary to force the search direction to have a negative inner product with each of the gradients. To be precise, the search direction is a blend of gradients and in-the-large directions and is computed as the solution to the following optimization problem.

Find the minimum 2-norm d that satisfies

$$\begin{aligned} \frac{-Dc_i(x)d}{\|Dc_i(x)\|_2} &\geq 1, \forall i \in V_c \\ \frac{-Dh_i(\omega_j, x)d}{\|Dh_i(\omega_j, x)\|_2} &\geq 1, \forall (i, j) \in V_f \\ \frac{r_i^T(x)d}{b_i \|r_i(x)\|_2} &\geq 1, \forall i \in V_r \end{aligned}$$

where the r_i are in-the-large directions and V_r is a set denoting the violated constraints for which the in-the-large directions are to be used. The scalars b_i are chosen to reflect the desired weighting between the in-the-large directions and their corresponding gradients. In general, the b_i are kept greater than one in order to place more weight on the in-the-large directions than the gradients.

The motivation for using an inequality constrained optimization problem to find the search direction is to reduce the angle between the desired directions (gradients and in-the-large directions) and the search direction. To illustrate why this approach is potentially better than simply solving a linear system, let $\{v_i \in \mathbb{R}^{N_v}, 1, 2, \dots, N_v\}$ be a set of vectors where $N_v \leq N_p$. If the search direction is calculated according to the method, i.e., find the d having the smallest 2-norm that satisfies

$$\frac{v_i^T d}{\|v_i\|_2} = 1, \quad i = 1, 2, \dots, N_v, \quad (4.50)$$

then the angle between the search direction and each v_i , denoted by $\theta(v_i, d)$ satisfies

$$\cos[\theta(v_i, d)] = \frac{1}{\|d\|_2}, \quad i = 1, 2, \dots, N_v. \quad (4.51)$$

On the other hand, if the inequality method, i.e., find the d having the smallest 2-norm that satisfies

$$\frac{v_i^T d}{\|v_i\|_2} \geq 1, \quad i = 1, 2, \dots, N_v, \quad (4.52)$$

is used, then the angle between the search direction and each v_i satisfies

$$\cos[\theta(v_i, d)] \geq \frac{1}{\|d\|_2}, \quad i = 1, 2, \dots, N_v. \quad (4.53)$$

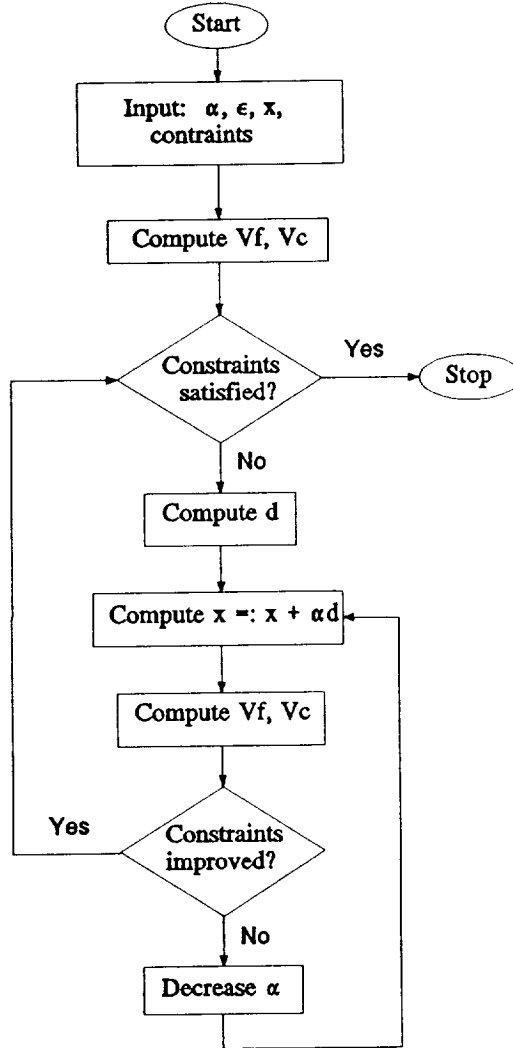


Figure 4.4 Simplified Flowchart of Algorithm A2

Thus the possibility exists that the search direction can be made closer (in direction) to each of the v_i through the use of the inequality method. This method is sometimes referred to as a *least distance program* (Lawson and Hanson, 1974) because it computes a search direction that takes the shortest path to a given region.

If the number of constraints in the search direction calculation is greater than the dimension of the parameter space, it is likely that the problem is ill-posed. In this case it is necessary to decrease the value of ϵ until the number of constraints is less than or equal to the number of parameters. Once the search direction is calculated, the step length is determined by requiring that the worst violation of every violated constraint decrease. A simplified flowchart of the algorithm is given in Figure 4.4.

The application of this algorithm to a controller design problem for the Hubble Space Telescope is presented in sub-section 4.8. As a result of that design effort a few weaknesses of the algorithm were discovered and are described therein.

4.4.3 Algorithm A3

This final algorithm, being the result of careful study of the conditions under which all the previously discussed algorithms tend to jam, has proven to be highly robust to jamming and usually able to make more rapid progress toward the feasible region than any of the algorithms discussed. The increased robustness and performance is achieved by making the search direction depend upon the anticipated step length in addition to the constraint function gradients. Thus, this algorithm differs from all the previously described algorithms in that it does not follow the Standard Model Algorithm, but uses the following model algorithm structure instead.

Anticipative Model Algorithm

Let $x \in R^N$ be the current parameter vector.

- Step 1. [Test for a solution.] If the conditions for a solution are satisfied, the algorithm terminates with x as the solution.
- Step 2. [Compute a search direction.] Compute a non-zero direction of search $d \in R^N$ that improves the linear

approximation to the problem assuming a step length of $\alpha \in R$.

- Step 3. [Check for an improvement.] Check if the measure of algorithm progress has improved. If not, decrease α and go to step 2.
- Step 4. [Update the parameter vector.] Set $x := x + \alpha d$ and go to step 1.

Note that this model algorithm differs from the Standard Model Algorithm in Steps 2 and 3.

To describe the complete algorithm in detail, first define the worst violation at each iteration by

$$\psi(x) = \max\{c_i(x), i = 1, 2, \dots, N_c; f_i(x), i = 1, 2, \dots, N_f\}. \quad (4.54)$$

Then for $\epsilon > 0$ denote the ϵ -active constraints as

$$V_\epsilon = \{i \mid c_i(x) > \psi(x) - \epsilon\} \quad (4.55)$$

and

$$V_f = \{(i, j) \mid h_i(\omega_j, x) > \psi(x) - \epsilon, \omega_j \in \Omega_i\}. \quad (4.56)$$

Calculate the search direction at each iteration by solving

$$\theta(x) = \min_{\|d\|_1 = \alpha} \max\{c_i(x) + Dc_i(x)d, i \in V_\epsilon; h_i(\omega_j, x) + Dh_i(\omega_j, x)d, (i, j) \in V_f\}. \quad (4.57)$$

Calculating the search direction in this manner is equivalent to finding a parameter correction of length α that affords the maximum decrease in the value of the worst constraint violation predicted by the first-order approximations to all the functions corresponding to the ϵ -active constraints. If the parameter correction calculated by Equation 4.57 fails to decrease the actual value of the worst violation, the step length α is decreased, *and the search direction is recalculated* using Equation 4.57 until

$$\psi(x + d) < \psi(x) - \alpha\beta\delta \quad (4.58)$$

where $\beta \in [0, 1)$ is chosen to guarantee a sufficient decrease (a value of 0.2 has been used with consistent success) and $\delta = \psi(x) - \theta(x)$. At the next iteration the step

length is increased by a user defined factor over the previous successful step length. The algorithm is terminated if all the constraints are satisfied or if the minimum step length allowed by the user is reached.

Several features of the above technique for calculating the search direction are now mentioned. First, the parameter ϵ is used to determine how many gradients are to be computed and not as an intrinsic part of the search direction calculation. Second, unlike all the previous algorithms, the search direction is calculated to achieve precisely what is actually desired according to the measure of constraint improvement

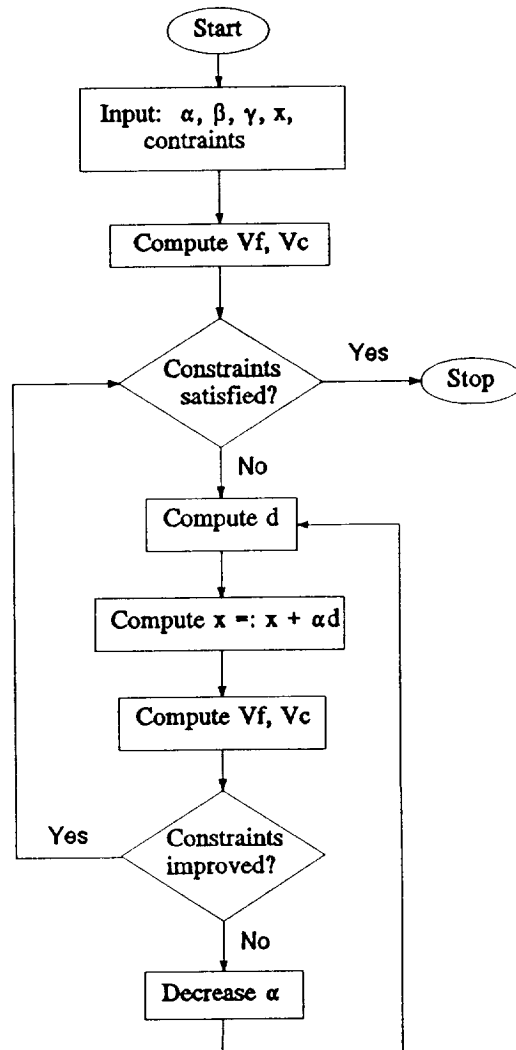


Figure 4.5 Simplified Flowchart of Algorithm A3

(the value of the worst violation). Recall that in the Polak-Mayne Algorithm the measure of improvement is also the value of the worst violation, but the search direction calculation does not explicitly take this into account. Third, this "look ahead" approach uses the proposed step length as a integral part of the search direction calculation. This is important because the step length is an implicit indicator of the quality of the first-order approximations of the constraint functions in the sense that if a large step length can be taken, the first-order approximations are usually reliable, whereas, if only a short step length is successful, the first-order approximations are not reliable. Fourth, if the initial correction fails to improve the actual constraints, then rather than simply reducing the step length until an improvement is registered, the search direction is also recalculated assuming the new step length. A simplified flowchart of the algorithm is given in Figure 4.5.

4.5 Controller Parameterizations

When applying search-based methods to controller design problems it is necessary to choose how the controller is to be represented. This aspect of search-based methods appears to be one of the least studied; yet, it can be an important factor in designing a practical algorithm because of the effect it has on the characteristics of the parameter space.

4.5.1 State-Space Realizations

A proper, p -input, q -output LTI controller K of order n can be defined by specifying an ordered quadruple of matrices (A, B, C, D) where $A \in R^{n \times n}$, $B \in R^{n \times p}$, $C \in R^{q \times n}$ and $D \in R^{q \times p}$. The quadruple (A, B, C, D) is usually referred to as a state-space or state-variable realization of K . This name comes from the representation of a LTI system as a system of linear differential equations (or difference equations in the discrete-time case), i.e.,

$$\begin{aligned} \frac{dx}{dt}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) + Du(t) \end{aligned} \quad , \quad (4.59)$$

where $x(t)$ is the state vector, $u(t)$ is the input vector, and $y(t)$ is the output vector. Given a state-space realization, the transfer function matrix of K can be calculated from

$$K(s) = C(sI - A)^{-1}B + D \quad , \quad (4.60)$$

where $s = j2\pi f$ for continuous-time systems, or $s = e^{j2\pi fT}$, (T_s is the sample period) for discrete-time systems.

An important property of state-space realizations is that for a given K the quadruple (A, B, C, D) is not uniquely determined. In fact, given any invertible T the controller defined by $(T^{-1}AT, T^{-1}B, CT, D)$ is the same as the controller defined by (A, B, C, D) . This fact, which is easily proved by direct substitution into Equation 4.60, reveals that any particular state-space realization of a given controller is simply one element of the equivalence class of all realizations of that controller. This raises the question of what effect the particular choice of realization has upon the dimension (number of scalar parameters) and topology (continuity, convexity, etc.) of the parameter space. This issue is an important area for future research.

A controller defined by an arbitrary quadruple (A, B, C, D) has $n^2 + n(p + q) + pq$ scalar parameters; therefore, the dimension of the parameter space increases rapidly with increasing controller order n . One way to reduce the dimension is to transform an arbitrary realization to one in which A has only n nontrivial parameters (trivial parameters are defined to be either 1 or 0), leaving $n + n(p + q) + pq$ parameters for design. Unfortunately, the transformation to such a form, often referred to as a *companion form*, can be numerically ill-conditioned (Golub and Van Loan, 1989, p. 369).

Recently, a technique for parameterizing minimal controller realizations of a fixed order in terms of B , C , and D ($n(p + q) + pq$ parameters) has been developed (Davis, Collins, and Hodel, 1992). The transformation of an arbitrary realization to the corresponding realization, referred to as the *input-normal Riccati form*, tends to be better conditioned than the transformation to a companion form. A possible difficulty in using this form, however, is that it is not defined for nonminimal realizations.

One compromise to the above approaches for reducing the number of design parameters is to transform the state-space into a form in which A is in *real-Schur form* (Golub and Van Loan, 1989, p. 362). This is a quasi-upper triangular matrix form that has all its real eigenvalues on the main diagonal and its complex eigenvalues in 2-by-2 blocks on the main diagonal. In addition to its eigenvalue revealing properties, the real-Schur form has the advantage that the similarity transformation into this form is always orthogonal. This means, among other things, that the transformation is almost always numerically well-conditioned. Another useful property is that the eigenvalues are independent of many of the elements above the main diagonal. Thus,

changing these elements can only modify the zeros of the controller. Experience in this research effort has revealed that the following parameterization strategy employing the real-Schur form can be effective.

- 1) Convert the initial controller state-space realization to a realization such that A is in real-Schur form.
- 2) Use the entries of the diagonal and first subdiagonal of A and all of the entries of B , C , and D as design parameters.
- 3) After each successful step of the algorithm, convert the updated realization to a realization such that A is in real-Schur form.

This technique uses $2n - 1 + n(p + q) + pq$ parameters.

The final parameterization to be described has been devised to allow for complete control over controller eigenvalues and to provide for a low number of design parameters. It is similar to the input-normal Riccati form in that it uses only the elements of B , C , and D for parameterization. This is accomplished by embedding B into A using the realization $(A_F + BF, B, C, D)$, where $A = A_F + BF$, and $F \in R^{p \times n}$ is chosen such that the pair (A_F, F) has certain desirable characteristics discussed below. This parameterization shall be referred to as the *state feedback parameterization* because of its resemblance to a linear system that has been compensated by a constant state feedback gain matrix F (Kailath, 1980, p. 500).

The question arises as to what controllers can be realized using this parameterization assuming that A_F and F are fixed and B , C , and D are free to vary. This question is answered by assuming an arbitrary realization $(\bar{A}, \bar{B}, \bar{C}, \bar{D})$ and finding the conditions for which $(\bar{A}, \bar{B}, \bar{C}, \bar{D})$ and $(A_F + BF, B, C, D)$ are equivalent. The previous discussion on state-space realizations indicates that these realizations are equivalent if there exist B and C and an invertible T such that

$$\bar{A}T = T(A_F + BF) , \quad (4.61)$$

$$\bar{B} = TB , \quad (4.62)$$

and

$$\bar{C}T = C . \quad (4.63)$$

Substituting Equation 4.62 into Equation 4.61 yields

$$\bar{A}T - TA_F = \bar{B}F . \quad (4.64)$$

This linear matrix equation is in the form of a Sylvester equation (Golub and Van Loan, 1989, p. 387) and can always be solved for T , but not necessarily guaranteed to have a full rank solution, if no eigenvalue of \bar{A} is equal to any eigenvalue of A_F (a sufficient condition). Once T is obtained, C is given by Equation 4.63 and B is given by

$$B = T^{-1}\bar{B} . \quad (4.65)$$

The authors have not been able to determine the conditions for the existence of a full rank solution to Equation 4.64; however, this parameterization has been used in many design experiments during this research effort with no difficulties.

The choice of the matrix F is where the utility of this parameterization is fully realized. By judicious choice of F it is possible to keep some eigenvalues of A fixed while allowing others to vary. This is highly desirable since it is quite common to require particular poles of the controller to have certain values in order that the closed-loop control system possess specific properties. As applied to this particular problem, this implies that to keep a particular eigenvalue of A fixed at an eigenvalue of A_F , all the rows of F should be chosen orthogonal to the corresponding eigenvector of A_F (Anderson and Moore, 1990, p. 354). As a simple example, if it is desired to keep all of the eigenvalues of A fixed to those of A_F , F can be chosen as the zero matrix.

4.5.2 Rational Function Representations

Another way to parameterize a controller is to represent each element of a controller transfer function matrix as a rational function. For example, each element can be defined either in the form

$$K(s) = K_0 \frac{s^n + a_{n-1}s^{n-1} + \dots + a_1s + a_0}{s^n + b_{n-1}s^{n-1} + \dots + b_1s + b_0} \quad (4.66)$$

or

$$K(s) = K_0 \frac{\prod_{i=1}^{n/2} (s^2 + a_{i1}s + a_{i2})}{\prod_{i=1}^{n/2} (s^2 + b_{i1}s + b_{i2})} . \quad (4.67)$$

Empirical evidence suggests that the factored form of Equation 4.67 provides better performance in mathematical programming algorithms than the form of Equation 4.66

(Mitchell, 1972). This may be because the individual poles and zeros of each element of the transfer function depend only on a few parameters rather than all the coefficients of a polynomial. The transfer function representation has the disadvantage of not allowing a simple way of fixing the order of the controller while also allowing sufficient freedom to realize a wide class of controller structures. Because of this inflexibility, further properties of this type of controller representation are not considered.

4.6 Partial Derivative Calculations

In sub-section 4.1 a brief description of a few common control system design constraints was given. In this sub-section techniques for calculating these constraints and their partial derivatives with respect to the controller design parameters discussed in sub-section 4.5 are developed. The analytical expressions for the partial derivatives of the constraint functions are needed by the algorithms used in search-based methods for controller design in order to accurately determine the local behavior of the constraint functions. Experience has shown that the use of finite differences to obtain approximations to partial derivatives often leads to poor performance in mathematical programming (Gill, Murray, and Wright, 1981, p. 127).

4.6.1 Controller Frequency Response

In order to calculate the partial derivatives of frequency response constraint functions, it is helpful to first compute general expressions for the partial derivatives of the frequency response of a controller (at a fixed frequency) with respect to the design parameters. Let (A, B, C, D) be a state-space realization of a controller K . Then the frequency response of K at s is given by

$$K(s) = C\Phi(s)B + D \quad (4.68)$$

where $\Phi(s) = (sI - A)^{-1}$, and $s = j2\pi f$ for a continuous-time controller, or $s = e^{j2\pi fT}$ for a discrete-time controller. Define e_i as column vector that has a 1 as the i^{th} element with all other elements zero. Then the (i, j) element of $K(s)$ is given by

$$K_{ij}(s) = e_i^T [C\Phi(s)B + D] e_j. \quad (4.69)$$

Taking the partial derivative of $K_{ij}(s)$ with respect to the (m, n) element of D yields

$$\frac{\partial K_{ij}(s)}{\partial D_{mn}} = e_i^T e_m e_n^T e_j, \quad (4.70)$$

where the fact that

$$\frac{\partial D}{\partial D_{mn}} = e_m e_n^T \quad (4.71)$$

has been used. Rewriting Equation 4.70 reveals that

$$\frac{\partial K_{ij}(s)}{\partial D_{mn}} = e_n^T e_j e_i^T e_m, \quad (4.72)$$

which implies that

$$\frac{\partial K_{ij}(s)}{\partial D} = [e_j e_i^T]^T. \quad (4.73)$$

Using a similar approach it can be shown that

$$\frac{\partial K_{ij}(s)}{\partial C} = [\Phi(s) B e_j e_i^T]^T, \quad (4.74)$$

$$\frac{\partial K_{ij}(s)}{\partial B} = [e_j e_i^T C \Phi(s)]^T, \quad (4.75)$$

and

$$\frac{\partial K_{ij}(s)}{\partial A} = [\Phi(s) B e_j e_i^T C \Phi(s)]^T. \quad (4.76)$$

Equations 4.73-4.76 assume that all the elements of A , B , C , and D are independent. If A is a function of B as in the state feedback parameterization

$$A = A_F + BF, \quad (4.77)$$

then Equation 4.76 is eliminated and Equation 4.75 is replaced with

$$\frac{\partial K_{ij}(s)}{\partial B} = \{[I + F\Phi(s)B]e_j e_i^T C \Phi(s)\}^T. \quad (4.78)$$

Since the state-space realization of a given controller is not unique, as pointed out in sub-section 4.5, it is interesting to consider how the particular choice of realization changes partial derivative calculations. Consider a controller K parameterized by the realization (A, B, C, D) and the realization (A_T, B_T, C_T, D_T) , where $A_Q = Q^{-1}AQ$, $B_Q = Q^{-1}B$, $C_Q = CQ$, and $D_Q = D$. Direct substitution of

these equalities into Equations 4.73-4.76, yields, in terms of the original partial derivatives, the following expressions:

$$\frac{\partial K_{ij}(s)}{\partial C_Q} = \frac{\partial K_{ij}(s)}{\partial C} Q^{-T}, \quad (4.79)$$

and
$$\frac{\partial K_{ij}(s)}{\partial B_Q} = Q^T \frac{\partial K_{ij}(s)}{\partial B}, \quad (4.80)$$

$$\frac{\partial K_{ij}(s)}{\partial A_Q} = Q^T \frac{\partial K_{ij}(s)}{\partial A} Q^{-T}. \quad (4.81)$$

Since D_T does not depend on T , $\frac{\partial K_{ij}(s)}{\partial D_T} = \frac{\partial K_{ij}(s)}{\partial D}$. These results indicate that the choice of realization changes the partial derivatives; but more importantly, they reveal that the transformation does not operate on the partial derivatives in the same manner as it operates on the original realization. This implies that the behavior of the constraint functions can be altered by changing the state-space realization of the controller. How to use this fact in developing algorithms for search-based controller design remains an area open to further research. For example, one possibility might be to try to determine the realization that minimizes some measure of the sensitivity of the satisfied constraints to the design parameters in order to reduce the likelihood that satisfied constraints become unsatisfied as the violated constraints are improved.

4.6.2 Frequency Response of a Transfer Function

General expressions for the partial derivatives of a controller frequency response with respect to the controller design parameters were derived in the previous section. In order to calculate the partial derivatives of constraint functions, such as frequency dependent singular values, frequency response magnitudes of individual elements of a transfer function matrix, operator norms, etc., it is necessary to develop general expressions for the partial derivatives of the frequency response of a transfer function matrix with respect to the controller design parameters.

Let $T: C^{p \times q} \rightarrow C^{r \times s}$ be a function of the frequency response of a controller (not the design parameters) evaluated at a single frequency as defined by

$$T(K) = T_R(K) + jT_I(K) \quad (4.82)$$

where K is the frequency response of the controller, and $T_R(K) \in \mathbb{R}^{r \times s}$ and $T_I(K) \in \mathbb{R}^{r \times s}$ are the real and imaginary parts of $T(K)$, respectively. Define $\frac{\partial T}{\partial K_{ij}}(K) \in \mathbb{C}^{r \times s}$ as a matrix with its (m,n) element given by $\frac{\partial T_{mn}}{\partial K_{ij}}(K)$, where K_{ij} is the (i,j) element of K . Let $K: \mathbb{R}^N \rightarrow \mathbb{C}^{p \times q}$ be a function of the real-valued vector \mathbf{x} (e.g., particular elements of a state-space realization). Now define $\frac{\partial K}{\partial x_r}(\mathbf{x}) \in \mathbb{C}^{p \times q}$ as a matrix with its (i,j) element given by

$$\frac{\partial K_{ij}}{\partial x_r}(\mathbf{x}) \triangleq \frac{\partial (K_R)_{ij}}{\partial x_r}(\mathbf{x}) + j \frac{\partial (K_I)_{ij}}{\partial x_r}(\mathbf{x}) \quad (4.83)$$

where $K_R = \text{Re}(K)$ and $K_I = \text{Im}(K)$. Now by a straightforward application of the multivariable chain rule it can be shown that for every (m,n) the partial derivatives of the real and imaginary parts of T with respect to the parameter x_r are given by

$$\frac{\partial (T_R)_{mn}}{\partial x_r} = \sum_{j=1}^q \sum_{i=1}^p \left[\frac{\partial (T_R)_{mn}}{\partial (K_R)_{ij}} \frac{\partial (K_R)_{ij}}{\partial x_r} + \frac{\partial (T_R)_{mn}}{\partial (K_I)_{ij}} \frac{\partial (K_I)_{ij}}{\partial x_r} \right], \quad (4.84)$$

and

$$\frac{\partial (T_I)_{mn}}{\partial x_r} = \sum_{j=1}^q \sum_{i=1}^p \left[\frac{\partial (T_I)_{mn}}{\partial (K_R)_{ij}} \frac{\partial (K_R)_{ij}}{\partial x_r} + \frac{\partial (T_I)_{mn}}{\partial (K_I)_{ij}} \frac{\partial (K_I)_{ij}}{\partial x_r} \right]. \quad (4.85)$$

These expressions can be reduced to a more compact form by first noting that application of the Cauchy-Riemann equations (Appendix B) to the function T_{mn} yields

$$\frac{\partial (T_R)_{mn}}{\partial (K_R)_{ij}} = \frac{\partial (T_I)_{mn}}{\partial (K_I)_{ij}}, \quad (4.86)$$

$$\frac{\partial (T_R)_{mn}}{\partial (K_I)_{ij}} = -\frac{\partial (T_I)_{mn}}{\partial (K_R)_{ij}}, \quad (4.87)$$

and

$$\frac{\partial T_{mn}}{\partial K_{ij}} = \frac{\partial (T_R)_{mn}}{\partial (K_R)_{ij}} + j \frac{\partial (T_I)_{mn}}{\partial (K_R)_{ij}}; \quad (4.88)$$

and then substituting these equations into Equations 4.84 and 4.85 to obtain

$$\frac{\partial T}{\partial x_r}(\mathbf{x}) = \sum_{j=1}^q \sum_{i=1}^p \frac{\partial T}{\partial K_{ij}}(K(\mathbf{x})) \frac{\partial K_{ij}}{\partial x_r}(\mathbf{x}). \quad (4.89)$$

The term $\frac{\partial T}{\partial K_{ij}}(K(x))$ in Equation 4.89 is, in general, different for various transfer function matrices. As an example of this calculation, consider the transfer function defined by Equation 4.1, which is repeated here for convenience:

$$T = (I + G_p K)^{-1} G_d . \quad (4.90)$$

Using the vectors e_i and e_j as defined in Section 4.6.1

$$\frac{\partial T}{\partial K_{ij}} = \frac{\partial}{\partial K_{ij}} [(I + G_p K)^{-1}] G_d , \quad (4.91)$$

$$\frac{\partial T}{\partial K_{ij}} = -(I + G_p K)^{-1} G_p \frac{\partial K}{\partial K_{ij}} (I + G_p K)^{-1} , \quad (4.92)$$

and

$$\frac{\partial T}{\partial K_{ij}} = -(I + G_p K)^{-1} G_p e_i e_j^T (I + G_p K)^{-1} . \quad (4.93)$$

It is interesting to note that this partial derivative is simply an outer product of vectors. In fact, this holds for all such partials, and it is important to use this information upon actual implementation in order to reduce floating-point operations.

In order to complete the development of the expressions necessary for evaluating constraint function gradients, the partial derivatives of several real-valued functions of the frequency responses of transfer functions are derived in the following sub-sections.

4.6.3 Frequency Dependent Singular Values

Sub-section 4.1 pointed out that many common performance and stability robustness constraints are defined in terms of the frequency dependent singular values of the frequency response of a transfer function. Since at any given frequency the frequency response of a transfer function is a complex-valued matrix, the general expressions for the necessary partial derivatives can be determined by calculating the partial derivatives of the singular values of a matrix with respect to some arbitrary scalar parameter upon which the elements of the matrix depend. Before expressions for these partial derivatives are derived, the basic properties of the SVD are presented.

Let $T \in C^{m \times n}$. Then there exist unitary matrices

$$U = [u_1 \ u_2 \ \dots \ u_m] \in C^{m \times m} \quad (4.94)$$

and

$$V = [v_1 \ v_2 \ \dots \ v_n] \in C^{n \times n} \quad (4.95)$$

such that

$$T = U \Sigma V^H, \quad (4.96)$$

where

$$\Sigma = \text{diag}(\sigma_1, \sigma_2, \dots, \sigma_q) \in R^{m \times n}, \quad q = \min\{m, n\} \quad (4.97)$$

and $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_q \geq 0$. The σ_i are the singular values of T , and the u_i and v_i are the corresponding left singular vectors and right singular vectors, respectively. It is often useful to define $\sigma_{\max} = \sigma_1$ and $\sigma_{\min} = \sigma_q$. Several algorithms exist for the calculation of the SVD of a matrix. A well- tested implementation of an algorithm for calculating the SVD is provided in the LINPACK collection of FORTRAN subroutines (Dongarra *et al.*, 1979).

To calculate the partial derivatives of the i^{th} singular value of T with respect to the elements of some vector of parameters x , first note that Equation 4.96 implies that

$$T v_i = \sigma_i u_i \quad (4.98)$$

and

$$T^H u_i = \sigma_i v_i, \quad (4.99)$$

where $i = 1, 2, \dots, q$. Taking derivatives with respect to the j^{th} element of x by applying the product rule to Equations 4.98 and 4.99 yields (dropping the explicit dependence on x for the sake of brevity)

$$\frac{\partial T}{\partial x_j} v_i + T \frac{\partial v_i}{\partial x_j} = \frac{\partial \sigma_i}{\partial x_j} u_i + \sigma_i \frac{\partial u_i}{\partial x_j} \quad (4.100)$$

and

$$\frac{\partial T^H}{\partial x_j} u_i + T^H \frac{\partial u_i}{\partial x_j} = \frac{\partial \sigma_i}{\partial x_j} v_i + \sigma_i \frac{\partial v_i}{\partial x_j}, \quad (4.101)$$

respectively. Multiplying Equation 4.100 on the left by u_i^H and using the complex-conjugate transpose of Equation 4.99 yields

$$u_i^H \frac{\partial T}{\partial x_j} v_i + \sigma_i v_i^H \frac{\partial v_i}{\partial x_j} = \frac{\partial \sigma_i}{\partial x_j} + \sigma_i u_i^H \frac{\partial u_i}{\partial x_j}. \quad (4.102)$$

Likewise, multiplication of Equation 4.101 on the right by v_i^H and using the complex-conjugate transpose of Equation 4.100 yields

$$v_i^H \frac{\partial T^H}{\partial x_j} u_i + \sigma_i u_i^H \frac{\partial u_i}{\partial x_j} = \frac{\partial \sigma_i}{\partial x_j} + \sigma_i v_i^H \frac{\partial v_i}{\partial x_j}. \quad (4.103)$$

Adding Equations 4.102 and 4.103 and solving for $\frac{\partial \sigma_i}{\partial x}$ yields the desired result:

$$\frac{\partial \sigma_i(x)}{\partial x_j} = \text{Re} \left[u_i^H \frac{\partial T}{\partial x_j}(x) v_i \right], \quad (4.104)$$

which is the same as obtained by Junkins and Kim (1990). In the case of a frequency response constraint, the matrix T becomes a function of frequency as well as the controller parameters; and therefore, the singular value partial derivatives also become functions of frequency.

As described in sub-section 4.1, many design constraints can be expressed explicitly in terms of the maximum singular value frequency response of a transfer function. Experience has revealed that great care must be taken when trying to improve a constraint involving maximum singular values, because in many circumstances the maximum singular value is not well isolated (meaning that the next largest singular value is near it in value). If this is the case, trying to improve the maximum singular value without regard for the other singular values may cause another singular value to become the "maximum". Furthermore, this new maximum may be a worse constraint violation than the first one. A simple solution to this potential difficulty is to try to simultaneously improve all the singular values that are near in value to the maximum singular value. Experience in this research effort has shown this approach to be highly effective in practice. A similar approach is also used when constraining minimum singular values.

4.6.4 Operator Norms

Although the infinity-norm of an operator is, in general, defined on an uncountable set (a frequency interval), it can be approximated for practical purposes by using a finite set Ω , i.e.,

$$\|T\|_{\infty} = \sup \{ \sigma_{\max}[T(j\omega)], \omega \in \mathbf{R} \} \approx \max \{ \sigma_{\max}[T(j\omega)], \omega_j \in \Omega \}. \quad (4.105)$$

This approximation may seem rather coarse, but in practice there is usually no difficulty in defining Ω such that there is a point near the actual value of ω at which the norm is achieved. The calculation of the partial derivatives of the infinity norm follow directly from the singular value partial derivatives, although care must be used

when the norm is nearly achieved at more than one frequency point. This difficulty is overcome by a technique similar to that used for nearly repeated singular values.

The calculation of the operator 2-norm is somewhat more difficult than the infinity-norm. In this case, an integral must be replaced by a finite sum, i.e.,

$$\|T\|_2 = \frac{1}{\pi} \left\{ \int_0^\infty \text{tr}[T^H(j\omega) T(j\omega)] d\omega \right\}^{1/2} \quad (4.106)$$

can be approximated by

$$\|T\|_2 \approx \left\{ \frac{1}{\pi} \sum_{i=1}^{N_r} w_i \text{tr}[T^H(j\omega_i) T(j\omega_i)] \right\}^{1/2}, \quad (4.107)$$

where the w_i are determined by the particular quadrature scheme used. The calculation of the partial derivatives depend on having an expression for the partial derivatives of the trace operator with respect to the elements of $T(j\omega)$. This is obtained through the relationship

$$\text{tr}[T^H(j\omega) T(j\omega)] = \sum_{m,n} \text{Re}[T_{mn}(j\omega)]^2 + \text{Im}[T_{mn}(j\omega)]^2. \quad (4.108)$$

Thus the partial derivatives of the trace with respect to the real and imaginary parts of the (i,j) element are given by

$$\frac{\partial}{\partial \text{Re}[T_{ij}(j\omega)]} \text{tr}[T^H(j\omega) T(j\omega)] = 2 \text{Re}[T_{ij}(j\omega)] \quad (4.109)$$

and

$$\frac{\partial}{\partial \text{Im}[T_{ij}(j\omega)]} \text{tr}[T^H(j\omega) T(j\omega)] = 2 \text{Im}[T_{ij}(j\omega)]. \quad (4.110)$$

4.6.5 Damping Ratios of Poles and Zeros

As was mentioned in sub-section 4.1, it is sometimes desirable to place constraints on the damping ratios of the poles and zeros of the controller in order to achieve certain types of stability and performance robustness. The damping ratio of a stable, complex pole or zero of a continuous-time system can be calculated from its corresponding location in the complex plane λ_c as

$$\zeta = \frac{-\text{Re}(\lambda_c)}{\left[\text{Re}^2(\lambda_c) + \text{Im}^2(\lambda_c)\right]^{1/2}} . \quad (4.111)$$

A straightforward calculation yields the partial derivatives of the damping ratio with respect to $\text{Re}(\lambda_c)$ and $\text{Im}(\lambda_c)$, which are given by

$$\text{and} \quad \frac{\partial \zeta}{\partial \text{Re}(\lambda_c)} = \frac{-\text{Im}^2(\lambda_c)}{\left[\text{Re}^2(\lambda_c) + \text{Im}^2(\lambda_c)\right]^{3/2}} \quad (4.112)$$

$$\frac{\partial \zeta}{\partial \text{Im}(\lambda_c)} = \frac{\text{Re}(\lambda_c) \text{Im}(\lambda_c)}{\left[\text{Re}^2(\lambda_c) + \text{Im}^2(\lambda_c)\right]^{3/2}} . \quad (4.113)$$

In the case of discrete-time linear systems, damping ratios can be defined through the relationship

$$\lambda_c = \frac{1}{T_s} \ln(\lambda_d) , \quad (4.114)$$

where T_s is the sampling period, λ_c is the continuous-time root location, and λ_d is the discrete-time root location. In order to calculate the necessary partial derivatives, the derivative of Equation 4.114 with respect to λ_d is needed and is given by

$$\frac{\partial \lambda_c}{\partial \lambda_d} = \frac{1}{T_s \lambda_d} . \quad (4.115)$$

Applying the Cauchy-Riemann equations (see Appendix B) to Equation 4.115 yields the following expressions for the partial derivatives of $\text{Re}(\lambda_c)$ and $\text{Im}(\lambda_c)$ with respect to $\text{Re}(\lambda_d)$ and $\text{Im}(\lambda_d)$:

$$\text{and} \quad \frac{\partial \text{Re}(\lambda_c)}{\partial \text{Re}(\lambda_d)} = \frac{\partial \text{Im}(\lambda_c)}{\partial \text{Im}(\lambda_d)} = \text{Re} \left[\frac{\partial \lambda_c}{\partial \lambda_d} \right] \quad (4.116)$$

$$\frac{\partial \text{Im}(\lambda_c)}{\partial \text{Re}(\lambda_d)} = -\frac{\partial \text{Re}(\lambda_c)}{\partial \text{Im}(\lambda_d)} = \text{Im} \left[\frac{\partial \lambda_c}{\partial \lambda_d} \right] . \quad (4.117)$$

Using the chain rule, together with Equations 4.112, 4.113, 4.116, and 4.117 results in the final expressions for the partial derivatives of the damping ratio of a complex pole or zero of a discrete-time system with respect to $\text{Re}(\lambda_d)$ and $\text{Im}(\lambda_d)$. They are

$$\frac{\partial \zeta}{\partial \text{Re}(\lambda_d)} = \frac{\partial \zeta}{\partial \text{Re}(\lambda_c)} \frac{\partial \text{Re}(\lambda_c)}{\partial \text{Re}(\lambda_d)} + \frac{\partial \zeta}{\partial \text{Im}(\lambda_c)} \frac{\partial \text{Im}(\lambda_c)}{\partial \text{Re}(\lambda_d)} \quad (4.118)$$

and

$$\frac{\partial \zeta}{\partial \text{Im}(\lambda_d)} = \frac{\partial \zeta}{\partial \text{Re}(\lambda_c)} \frac{\partial \text{Re}(\lambda_c)}{\partial \text{Im}(\lambda_d)} + \frac{\partial \zeta}{\partial \text{Im}(\lambda_c)} \frac{\partial \text{Im}(\lambda_c)}{\partial \text{Im}(\lambda_d)}. \quad (4.119)$$

4.6.6 Eigenvalues

To calculate the damping ratios of the poles of a controller with realization (A, B, C, D) the eigenvalues of A need to be determined. If $A \in \mathbb{R}^{n \times n}$, then the eigenvalues of A are defined by the set

$$\Lambda(A) = \{\lambda \in \mathbb{C} \mid \det(\lambda I - A) = 0\} \quad (4.120)$$

A well-tested implementation of the QR algorithm for calculating the eigenvalues of a matrix exists in the EISPACK collection of FORTRAN subroutines (Smith *et al.*, 1976).

Let $\lambda \in \Lambda(A)$. Then any $x \in \mathbb{C}^n$ such that

$$Ax = \lambda x \quad (4.121)$$

is a right eigenvector of A corresponding to λ . Similarly, any $y \in \mathbb{C}^n$ such that

$$y^H A = \lambda y^H \quad (4.122)$$

is a left eigenvector of A corresponding to λ . Since the implementation of an algorithm for the simultaneous calculation of the right and left eigenvectors corresponding to a particular eigenvalue is not readily available, the following (not necessarily efficient) approach using the SVD can be used. Let $A_\lambda = \lambda I - A$ and let $A_\lambda = U \Sigma V^H$ be the SVD of A_λ . Then for every i such that $\sigma_i = 0$, v_i and u_i are left and right eigenvectors of A corresponding to λ . Moreover, for every $i \neq j$, v_i and v_j are linearly independent, and u_i and u_j are linearly independent.

The partial derivatives of a matrix eigenvalue with respect to a scalar parameter can be derived in a manner similar to that of singular value partials. Suppose that the elements of A are functions of a vector of parameters p , and let p_i denote the i^{th} element of p . Taking derivatives on both sides of Equation 4.121 yields

$$\frac{\partial A}{\partial p_i} x + A \frac{\partial x}{\partial p_i} = \frac{\partial \lambda}{\partial p_i} x + \lambda \frac{\partial x}{\partial p_i} . \quad (4.123)$$

Multiplying on the left by y^H , using Equation 4.122, and solving for $\frac{\partial \lambda}{\partial p_i}$ yields the desired expression:

$$\frac{\partial \lambda}{\partial p_i}(p) = \frac{y^H \frac{\partial A}{\partial p_i}(p)x}{y^H x} , \quad (4.124)$$

which is equivalent to the results obtained by Golub and Van Loan (1989, p. 344).

4.6.7 Generalized Eigenvalues

To calculate the damping ratios of the zeros of the individual elements of a transfer function matrix the generalized eigenvalues of a pair of matrices must be determined. Let (A, B, C, D) be a state-space realization of a controller K . Define

$$G = \begin{bmatrix} A & -b_j \\ c_i & -d_{ij} \end{bmatrix}, \quad H = \begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix}, \quad (4.125)$$

where c_i is the i^{th} row of C , b_j is the j^{th} column of B , and d_{ij} is the (i, j) element of D . Then the zeros of the (i, j) element of K are given by the set (Kailath, 1980, p. 76)

$$\Lambda(G, H) = \{ \lambda \in \mathbb{C} \mid \det(\lambda H - G) = 0 \} . \quad (4.126)$$

A well-tested implementation of the QZ algorithm for determining generalized eigenvalues is available in the EISPACK collection of subroutines (Garbow *et al.*, 1977).

Suppose $\lambda \in \Lambda(G, H)$. Then any $x \in \mathbb{C}^n$ such that

$$Gx = \lambda Hx \quad (4.127)$$

is said to be a right eigenvector of the pair (G, H) corresponding to λ . Similarly, any $y \in \mathbb{C}^n$ such that

$$y^H G = \lambda y^H H \quad (4.128)$$

is said to be a left eigenvector of the pair (G, H) corresponding to λ . The left and right eigenvectors for the generalized eigenvalue problem can be determined by the same technique as is described for ordinary eigenvalues in Section 4.6.6

By using Equations 4.127 and 4.128, the partial derivatives of generalized eigenvalues can be determined in a manner analogous to that used for determining ordinary eigenvalue partials. Let G be a function of a vector of parameters p and let p_i be the i^{th} element of p . Then the partial derivative of λ with respect to p_i can be shown to be given by

$$\frac{\partial \lambda}{\partial p_i}(p) = \frac{y^H \frac{\partial G}{\partial p_i}(p)x}{y^H H x} . \quad (4.129)$$

4.7 Techniques for Obtaining an Initial Controller

Before the algorithms presented in sub-sections 4.3 and 4.4 can be applied to controller design problems, it is necessary to obtain an initial controller (parameter vector) that is close enough to a controller that satisfies the design constraints in order for the algorithm to have a reasonable chance of finding a satisfactory solution. There are two methods currently available for obtaining initial controllers: (1) classical graphical methods and (2) analytical synthesis methods.

The case in which a parametric model of the plant is available is somewhat easier to handle than the nonparametric model case because it is usually possible to rely on analytical synthesis methods, such as LQG and H_∞ -optimization, to design a controller that achieves a few design constraints. Then a search-based method can be applied to try to achieve the remaining constraints while maintaining those that were already satisfied by the analytical design. The only serious drawback to this approach is the fact that the analytical methods produce controllers that have a dynamical order that is greater than or equal to the plant model order. Occasionally these controllers can be simplified by model reduction techniques (Anderson and Moore, 1990, chap. 10) without seriously degrading the design constraints. After the model reduction is performed a search-based method can be applied in an attempt to recover the degraded constraints as well as achieve other constraints. This approach is employed in the applications presented in sub-section 4.9.

The case when the plant is described solely by a nonparametric model is more difficult because there does not exist any analytical design method that can be applied in order to get an initial controller that satisfies any design constraints. Presently, the only approach is to use graphical design procedures with frequency response estimates as was done in the application presented in Section 4.8.1. In the case of SISO plants and well decoupled MIMO plants, this is not an insurmountable task. On the other

hand, if the plant is MIMO and not well decoupled, then obtaining a good initial controller can become very tedious. In this case, one approach that may offer some hope for open-loop stable plants is to work with the Youla parameterization of all stabilizing controllers (Youla *et al.*, 1976). Without reviewing the details of this theory, let it be stated that if G is a proper and stable plant, then the controller given by

$$K = Q(I - GQ)^{-1} \quad (4.130)$$

results in a stable closed-loop system (negative feedback assumed) where Q is any proper and stable transfer function. Moreover, it can be shown that closed-loop transfer functions, such as the sensitivity function and complementary sensitivity function are affine (linear) functions of Q . This affine relationship simplifies the choice of an initial Q and thus the design of an initial K . Of course, the drawback to this approach is that it results in a nonparametric model of the controller as well as the plant! This is clear from Equation 4.130. Therefore, one system identification problem (the plant) is traded for another (the controller). However, it may be possible to simultaneously design Q and identify a model of a fixed order for K . This is an issue open to research.

4.8 Successful Applications

This sub-section describes the successful application of Algorithms A1 and A2 to actual design problems. The first application is to a large aerospace structure ground test facility (Frazier and Irwin, 1993), and the second is to the Hubble Space Telescope (Irwin, Lawrence, Glenn, and Frazier, 1993).

4.8.1 The ACES Facility

An illustration of the NASA Marshall Space Flight Center Active Control Technique Evaluation for Spacecraft (ACES) ground test facility is shown in Figure 4.6. The ACES facility is suitable for the study of line-of-sight (LOS) and vibration suppression control issues as pertaining to flexible aerospace structures. The primary element of the ACES facility, a spare Voyager magnetometer boom, is a lightly damped beam measuring approximately 45 feet in length.

The goal of the control system design is to maintain the reflected laser beam in the center of the antenna, subject to disturbances generated by the base excitation table (BET). This is to be accomplished by using the following actuators: Image Motion Compensation (IMC) gimbals, Advanced Gimbal System (AGS), Linear

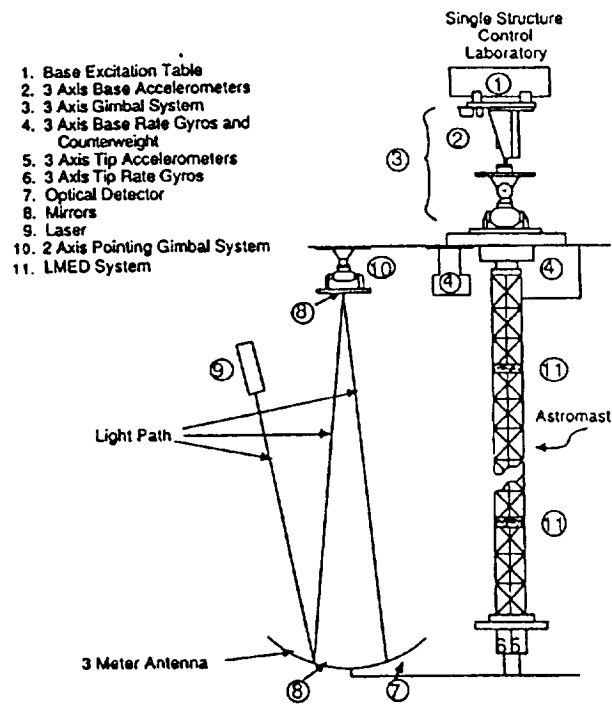


Figure 4.6 The ACES Ground Test Facility

Momentum Exchange Devices (LMED)'s; and sensors: base rate gyros (BGS), tip accelerometers, tip rate gyros, LMED positions and accelerations, and the optical position detector. As explained subsequently, this design only employed a subset of these sensors and actuators. The digital control law is to be implemented on the HP9000 control computer located at the facility using a fixed sampling rate of 50 Hertz and a fixed, one sample period computational delay.

The experimental open-loop frequency response from the y-axis IMC gimbal torque to the x-axis LOS error is shown in Figure 4.7, where the effect of the computational delay is quite apparent from analysis of the phase characteristic. The frequency responses of the other elements of the IMC-to-LOS transfer function matrix are similar, although the cross-axis terms have less gain. The open-loop frequency response from the y-axis AGS gimbal torque to the y-axis base rate gyro system (BGS) is shown in Figure 4.8, revealing the numerous lightly damped modes of the structure. The frequency responses of other elements of the AGS-to-BGS transfer function matrix are similar, but the cross axis terms have lower gain.

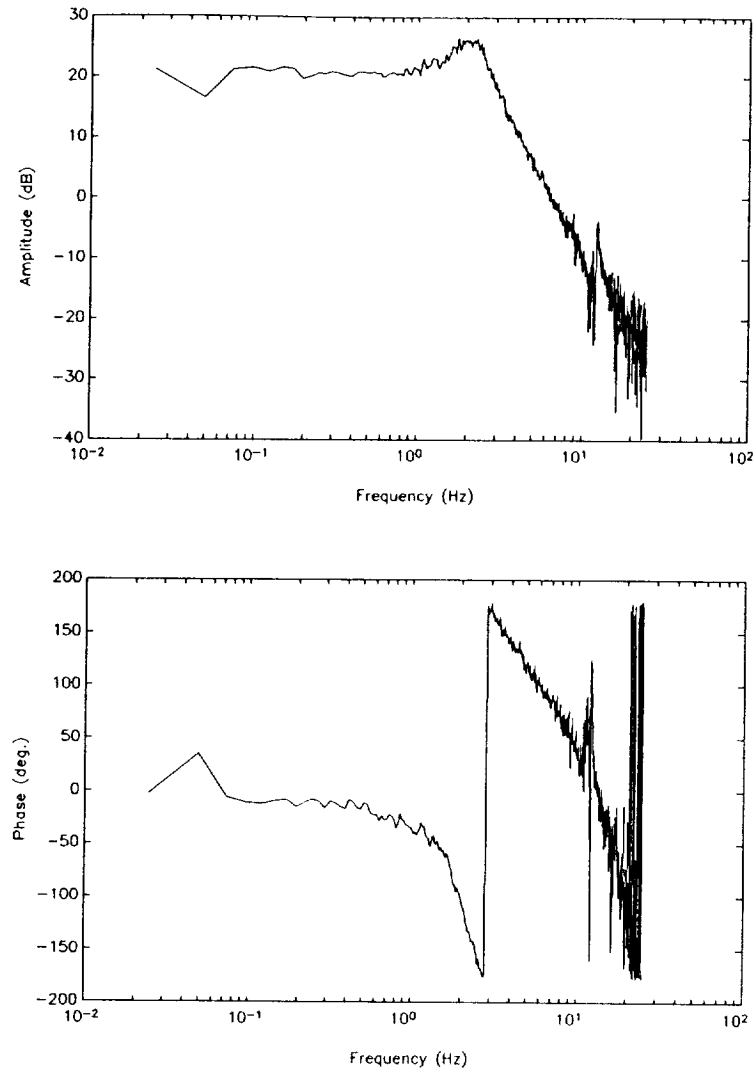


Figure 4.7 Experimental Frequency Response from y-axis IMC Gimbal to x-axis LOS Error

The philosophy taken for the design was to use the AGS and feedback from the BGS to add damping to the lightly damped modes of the beam and to use the IMC gimbals with feedback from the optical detector to maintain small LOS error. Since the IMC gimbals have very little impact on the boom, it is feasible to design controllers for the IMC-to-LOS and AGS-to-BGS subsystems independently. One concern, however, is the effect that the disturbances that reach the IMC gimbals through the connecting arm that is attached to the base (as opposed to disturbances acting on the detector directly) have on the LOS error, due to the inherently high optical gain from the IMC gimbals to the detector. One way to reduce the effects of

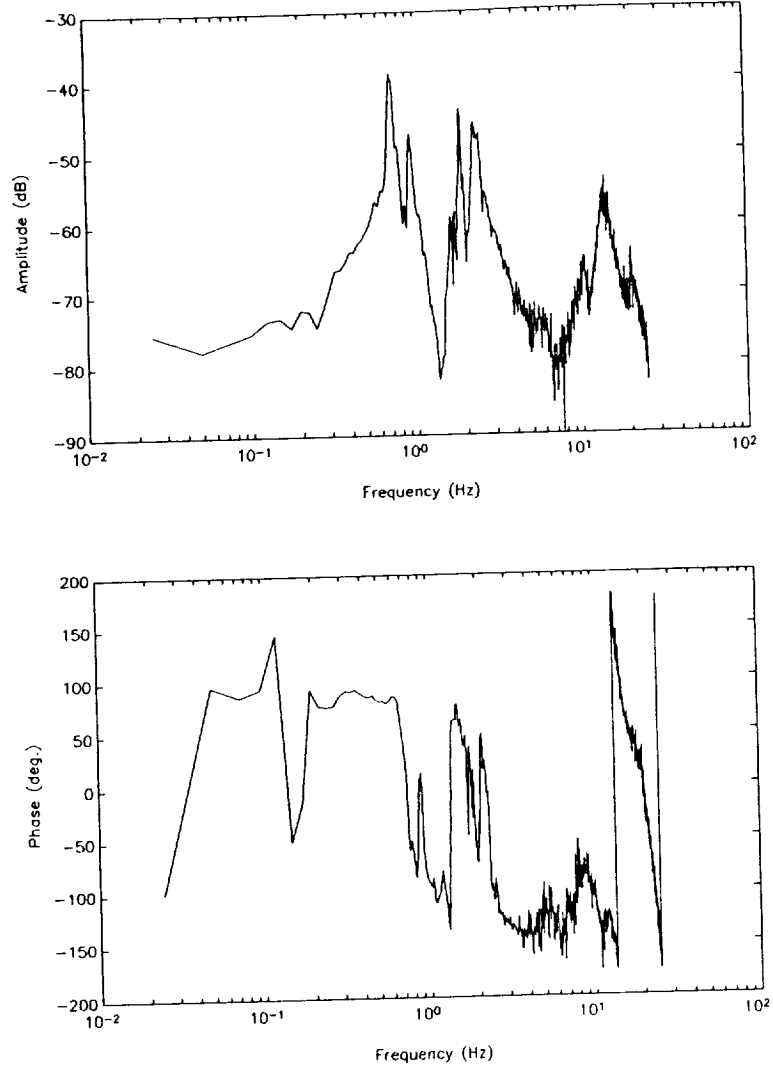


Figure 4.8 Experimental Frequency Response from y-axis AGS to y-axis Base Gyro

these disturbances is to maintain high IMC controller gain at the frequencies at which these disturbances occur. To understand why this is the case, consider the block diagram model of the IMC-to-LOS subsystem shown in Figure 4.9, where d_i represents the torque disturbances on the input to the IMC gimbals, d_o represents position disturbances on the detector, and y represents LOS error. The LOS error as a function of the disturbances is given by

$$y = (I + G_{IMC}K_{IMC})^{-1}d_o + (I + G_{IMC}K_{IMC})^{-1}G_{IMC}d_i. \quad (4.131)$$

The first term in Equation 4.131 reveals that high loop gain $G_{IMC}K_{IMC}$ provides good rejection of output position disturbances, but the second term reveals that if G_{IMC} has high gain (which it does in this case), then it is necessary for K_{IMC} to have high gain in order to have good rejection of input torque disturbances. Analysis of Figure 4.7 reveals that achieving high IMC controller gain while also maintaining acceptable stability margins is difficult because of the combination of the high optical gain and the additional phase lag introduced by the computational delay. Fortunately, the source of these disturbances can be reduced considerably by other means; namely, increasing the damping of the modes of the beam with the AGS, and thereby reducing the motion of the base and the arm supporting the IMC gimbals.

The next step of the design procedure was the determination of a set of precise closed-loop constraints that are consistent with the aforementioned design philosophy. These constraints are given in Table 4.1. Next, the initial controllers were designed for the IMC-to-LOS and AGS-to-BGS subsystems using standard, graphical one-loop-at-a-time techniques with the experimental frequency response data. Although the attempt was made to satisfy the design constraints when designing the initial controllers, they were not satisfied as can be seen by comparing the first and second columns of Table 4.1. The controller for each subsystem was 10^{th} order.

The multivariable design (i.e., taking cross-axis coupling within each subsystem into account) for each subsystem was then performed using the experimental data and Algorithm A1. The final values of all the constraint functions are provided in the third column of Table 4.1. Although most of the constraints were satisfied, two of the constraints for the AGS subsystem were not satisfied (the algorithm reached a point such that these constraints became locally unfeasible).

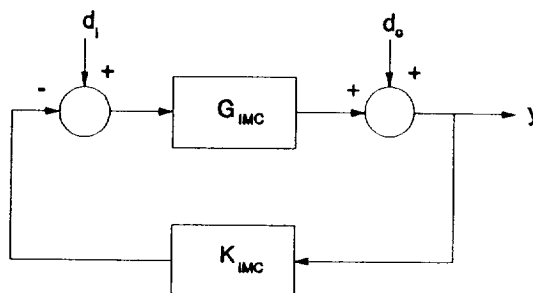


Figure 4.9 Block Diagram of the IMC-to-LOS Detector Subsystem

Table 4.1 Summary of Multivariable Design Constraint Values

Constraint	Initial Value	Final Value
$\sigma_{\min}[I + GK(z)]_{IMC} \geq 0.5, f \in (0,25)$	0.2289	0.5090
$\sigma_{\min}[I + KG(z)]_{IMC} \geq 0.5, f \in (0,25)$	0.2276	0.5056
$\sigma_{\min}[I + (GK(z))^{-1}]_{IMC} \geq 0.6, f \in (0,25)$	0.2827	0.6072
$\sigma_{\min}[I + (KG(z))^{-1}]_{IMC} \geq 0.6, f \in (0,25)$	0.2805	0.6112
$\sigma_{\min}[I + GK(z)]_{IMC} \geq 18, f \in [0.14, 0.16]$	10.0020	14.1000
$\sigma_{\min}[I + GK(z)]_{AGS} \geq 0.6, f \in (0,25)$	0.3649	0.5996
$\sigma_{\min}[I + KG(z)]_{AGS} \geq 0.6, f \in (0,25)$	0.3585	0.5988
$\sigma_{\min}[I + (GK(z))^{-1}]_{AGS} \geq 0.7, f \in (0,25)$	0.3600	0.6719
$\sigma_{\min}[I + (KG(z))^{-1}]_{AGS} \geq 0.7, f \in (0,25)$	0.3589	0.6712
IMC represents IMC subsystem; AGS represents AGS subsystem G represents plant; K represents controller $z = e^{j2\pi f T_s}, T_s = 0.02 \text{ sec}$		

However, it was decided to test the resulting controller anyway. To further illustrate results from the algorithm, Figures 4.10 and 4.11 show the resulting singular value frequency responses of $[I + GK]_{IMC}$ with the initial and final controllers, respectively. Note that these results were obtained from numerical manipulations and are not experimental closed loop responses.

After the design was completed, the resulting controller was implemented at the ACES facility. The open-loop x-axis LOS error due to an x-axis spacecraft crew motion BET disturbance is shown in Figure 4.12. The dominant behavior in the response is the lightly damped pendulum mode. The x-axis LOS error response to the same disturbance after closing the loop with the controller is shown in Figure 4.13. This figure illustrates significant improvement over the open-loop response. Experimental results for a y-axis BET disturbance also indicated performance improvements; however, they were not as significant as those obtained for the x-axis.

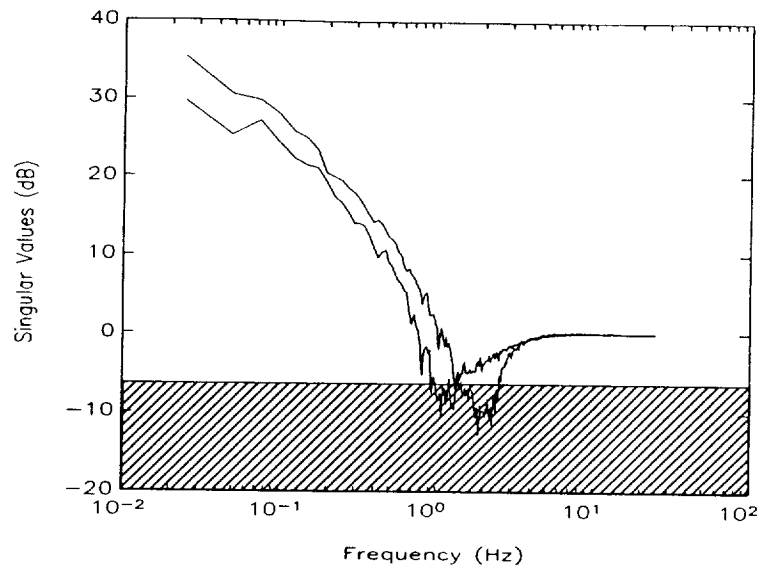


Figure 4.10 Initial Return Difference Singular Value Frequency Response (IMC)

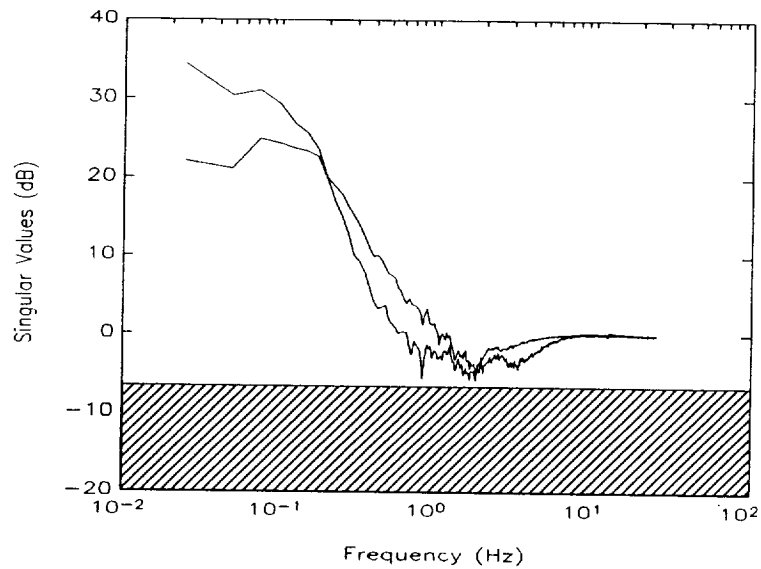


Figure 4.11 Final Return Difference Singular Value Frequency Response (IMC)

During the course of this design effort a major weakness in Algorithm A1 was discovered: the choice of the parameter N_{\max} had a significant impact on the performance of the algorithm. It was observed that if N_{\max} was too large, many gradients associated with constraint violations that were not even close to the worst violation for each frequency dependent constraint entered into the search direction

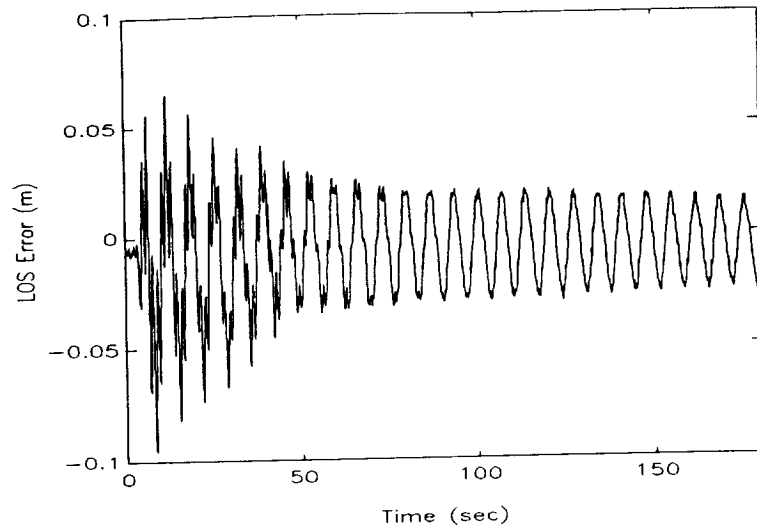


Figure 4.12 Open Loop LOS Error Transient Response

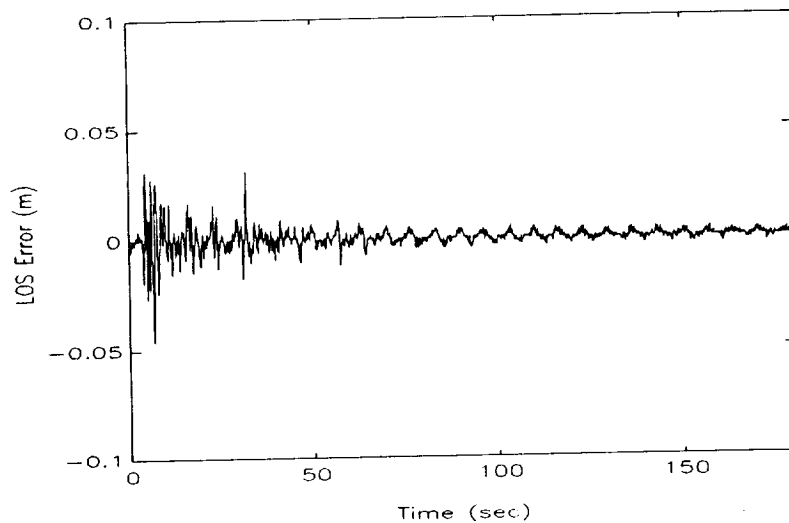


Figure 4.13 Closed Loop LOS Error Transient Response

calculation and thereby impeded progress. On the other hand, if N_{\max} was too small, too few gradients entered into the search direction calculation to prevent jamming. In order to get the algorithm to complete a successful design, this parameter had to be changed many times throughout program execution. A method for automatically changing N_{\max} based upon relationships between the constraint gradients may be able to alleviate this difficulty.

4.8.2 The Hubble Space Telescope

This section presents the application of Algorithm A2 to controller redesign for the pointing control system of the Hubble Space Telescope (HST). Initial redesign efforts are described by Sharkey, Nurre, Beals, and Nelson (1992). For the sake of brevity, only that part of the redesign effort performed by Ohio University that involves the use of Algorithm A2 are presented here. Further details are described by Irwin, Lawrence, Glenn, and Frazier (1993).

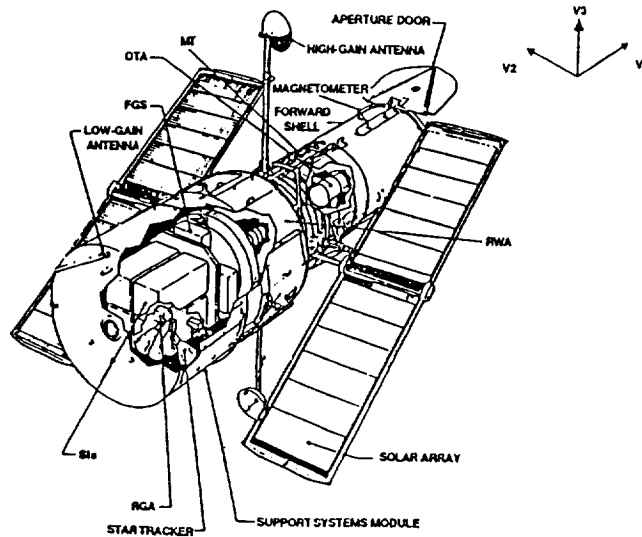


Figure 4.14 Illustration of the Hubble Space Telescope

An illustration of the HST is shown in Figure 4.14. The primary objective of the pointing control system is to maintain a fixed HST attitude so as to avoid pointing errors while using the telescope for scientific observation. Due to thermally induced vibrations in the solar arrays (SA's), the HST has encountered difficulties in maintaining satisfactory LOS pointing error requirements. Although efforts to reduce this error began shortly after the HST was placed into operational mode, satisfactory results have yet to be obtained.²

The effects of the SA vibrations are usually modeled as torque disturbances entering the plant as shown in the block diagram of Figure 4.15. Since the HST employs a sampled-data control system, the signals and transfer functions in Figure 4.15 are shown in the z-domain and are defined as

² At the time of the redesign, the HST repair mission had not yet flown.

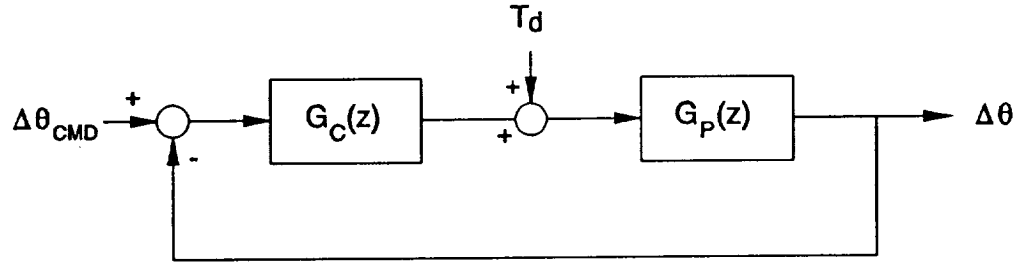


Figure 4.15 Block Diagram Used for HST Design

T_d : disturbance torque

$\Delta\theta_{CMD}$: commanded change in attitude

$\Delta\theta$: change in measured attitude

$G_p(z)$: discretized plant

$G_c(z)$: discrete-time controller

The change in measured attitude as a function of the disturbance torque is given by

$$\Delta\theta = (I + G_p G_c)^{-1} G_p T_d = S_o G_p T_d \quad (4.132)$$

where $S_o = (I + G_p G_c)^{-1}$ is the output sensitivity function.

Since it is desired to reduce the effect that T_d has on $\Delta\theta$, it is necessary to keep $S_o G_p$ small at frequencies where T_d contains significant power. Analysis of the power spectral densities of the attitude signals from the flight data indicates the presence of significant disturbance power between 0 Hz and 3 Hz, with large disturbance power levels occurring in the vicinity of 0.1 Hz and 0.6 Hz. The disturbance power at these two particular frequencies can be attributed to the

excitation of structural modes associated with the SA's. The 0.1 Hz mode is due to "out-of-plane" bending of the SA's and the 0.6 mode is due to "in-plane" bending. In order to obtain satisfactory performance, it is necessary to design the controller to attenuate disturbances across the entire frequency range from 0 Hz to 3 Hz, and not just the disturbances at 0.1 Hz and 0.6 Hz as has been suggested (Wie, Liu, and Bauer, 1993); although moderately damped controller poles near these frequencies can provide performance improvements without the loss of stability and performance robustness. Analysis of the geometry of the HST also indicates in-plane bending of the SA's should only have a minor impact on HST motion about the V_2 axis irrespective of SA orientation. On the other hand, in-plane bending can have a significant impact on the V_1 and V_3 axes, the degree of which depends on the particular SA orientation. The out-of-plane SA bending has an impact upon all HST axes, especially V_2 . These observations have been confirmed by analysis of the flight data.

Before describing the controller design that was performed using Algorithm A2, the SAGA-II controller (the best performing controller of those that have been flight tested) is analyzed in terms of the above observations. The SAGA-II controller was designed and implemented in order to achieve satisfactory pointing error attenuation in response to the SA generated disturbances. It is a diagonal controller consisting (in each axis) of second order PID compensation, lightly damped notch filters at 14 Hz or 19 Hz for "scissors mode" suppression, and lightly damped (3-9 percent damping) modes near 0.1 Hz and 0.6 Hz for improving disturbance rejection. The MIMO stability robustness problems with this controller are illustrated in Figure 4.16 by the small value (-11 dB) of the return difference near 1.5 Hz and in Figure 4.17 by the large value (12 dB) of the complementary sensitivity near 1.5 Hz. The singular value frequency response from the disturbance inputs to HST attitude is shown in Figure 4.18. This response indicates that significant disturbance power in the 0.2 Hz to 3 Hz range may have an undesirable effect upon the attitude error.

A 200 second linear simulation was performed using software and flight data provided by NASA corresponding to the 90 degree SA angle. The resulting peak and RMS values of the attitude error for all three axes are provided in Table 4.2. The corresponding PSD estimate of the V_3 axis is shown in Figure 4.19. The large peak near 1 Hz is a major contributor to the error in this axis.

Although system performance using the SAGA-II controller was greatly improved over the original PID controller, it was felt that further improvements could be obtained by modifying the SAGA-II controller using the following design

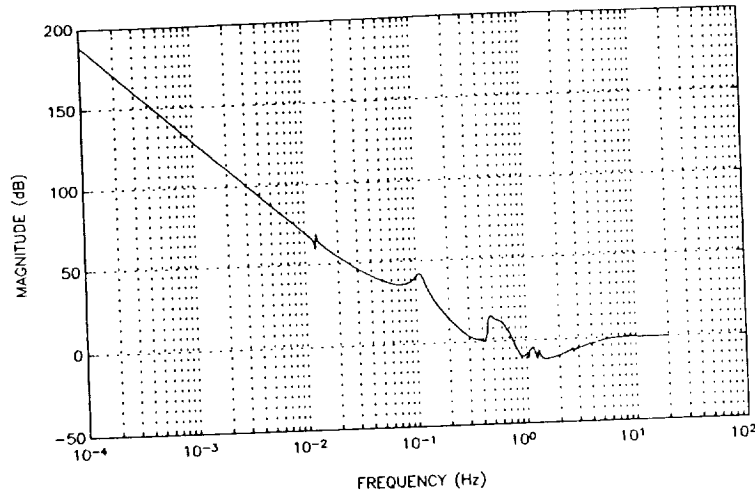


Figure 4.16 Return Difference Minimum Singular Value Frequency Response with SAGA-II Controller

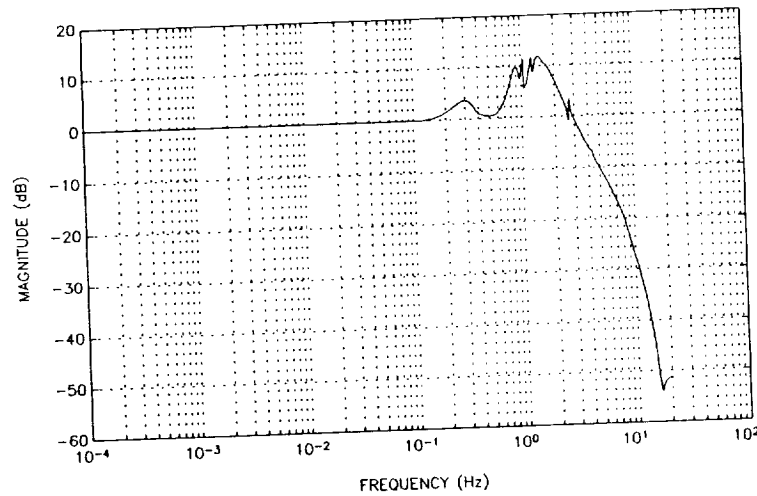


Figure 4.17 Complementary Sensitivity Maximum Singular Value Frequency Response with SAGA-II Controller

philosophy. The final controller should provide as much gain as possible from very low frequencies to approximately 3 Hz in order to reject torque disturbances. Special emphasis should be placed on increasing controller gain in the vicinity of 0.1 Hz in all three axes while increasing the gain in the vicinity of 0.6 Hz in the V_1 and V_3 axes. The controller should provide sufficient attenuation of modes at 14 Hz and 19 Hz.

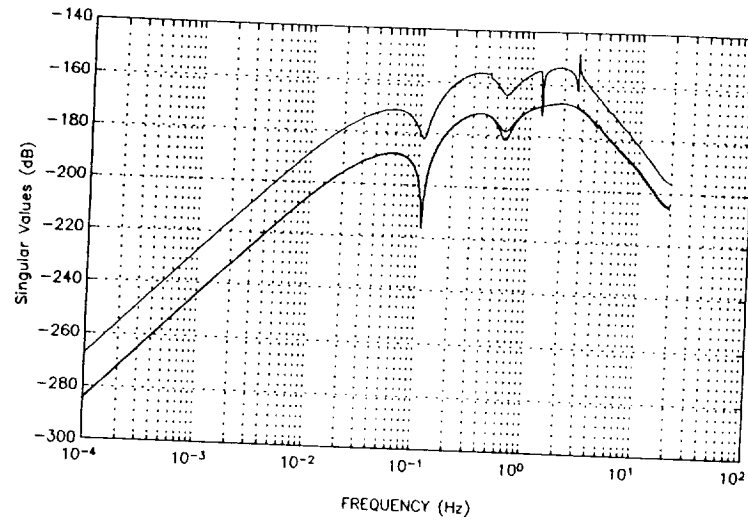


Figure 4.18 Singular Value Frequency Response from Disturbance to HST Attitude with SAGA-II Controller (All Three Singular Values Shown)

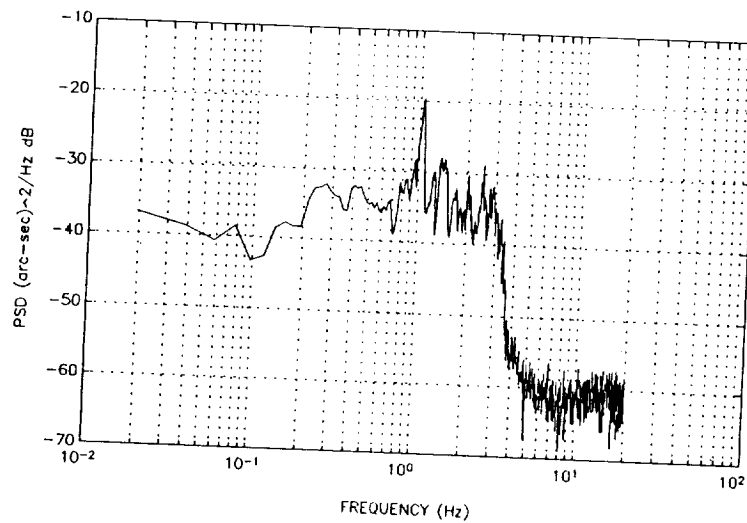


Figure 4.19 PSD Estimate of V_3 Axis Simulated Transient Response

Stability robustness should be specified in terms of the minimum singular value of the return difference operator and the maximum singular value of the complementary sensitivity function; both being MIMO measures of stability robustness to unstructured uncertainties.

Table 4.2 Peak and RMS Attitude Values for Linear Simulation with SAGA-II

	HST Axis		
	V_1	V_2	V_3
Peak Value (milliarcsec)	170.9	30.32	60.05
RMS Value (milliarcsec)	40.59	4.651	9.323

The initial controller for Algorithm A2 was designed as follows. To provide better performance robustness, the damping of all the lightly damped controller modes was increased to values ranging between 10 percent and 30 percent. In particular, the lightly damped mode at 0.63 Hz in the V_2 axis of the SAGA-II controller was eliminated based upon the aforementioned reasoning. This allowed for an increase in loop gain in this axis at lower frequencies, including 0.1 Hz. The PID stages and high frequency notch filters were unaltered. Additional second order lead compensation was added to each axis to help improve stability margins. Plots of the return difference minimum singular value and the complementary sensitivity maximum singular value are given, along with the desired specifications, in Figures 4.20 and 4.21, respectively. These plots indicate stability robustness problems near 3 Hz in both cases.

To improve stability robustness Algorithm A2 was employed. The PID stages, the high frequency notch filters, and the moderately damped controller modes were entered as fixed stages of compensation. The parameters of the second order compensation in each axis were used for design. The controller was parameterized with the real-Schur method described in sub-section 4.5.1. After approximately 300 steps the progress of the design had become sufficiently slow to warrant termination. Although significant progress had been made, the specifications had not been completely achieved. The final return difference minimum singular value and complementary sensitivity maximum singular value responses are shown in Figures 4.22 and 4.23, respectively. These figures indicate no apparent improvement in the return difference, but that significant improvement in the complementary sensitivity function was obtained. The singular value frequency response from the disturbances to HST attitude is shown in Figure 4.24. A comparison of this figure with Figure 4.18 reveals important reductions in the sensitivity of the HST attitude to disturbance

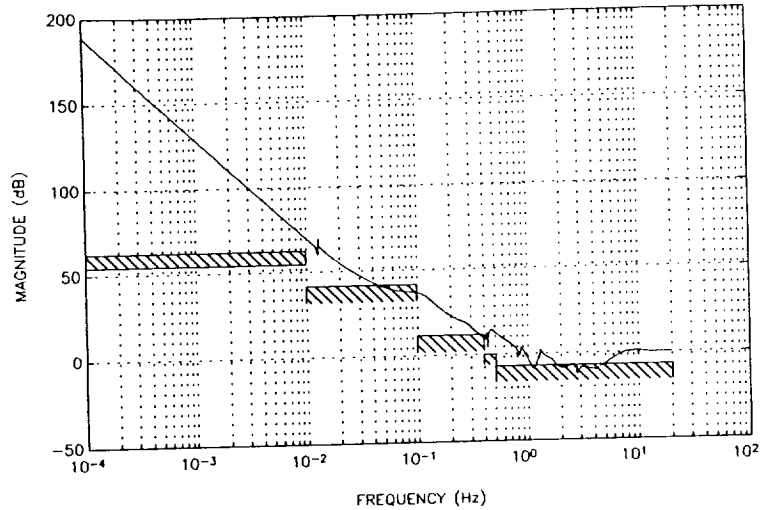


Figure 4.20 Return Difference Minimum Singular Value Frequency Response with Initial Controller

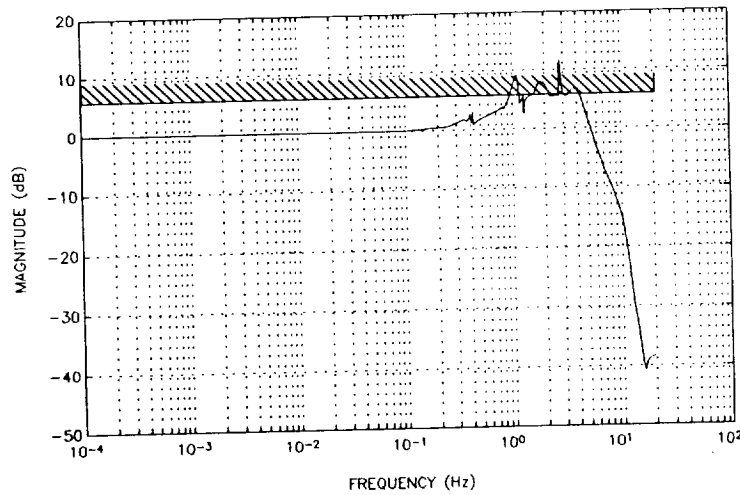


Figure 4.21 Complementary Sensitivity Maximum Singular Value Frequency Response with Initial Controller

power in the 0.2 Hz to 3 Hz range. Performing a 200 second linear simulation using the same disturbance as was used with SAGA-II resulted in the peak values and RMS values given in Table 4.3. These results indicate moderate performance improvements over the SAGA-II controller. These improvements can be attributed primarily to increased disturbance rejection at most frequencies below 3 Hz. The PSD estimate

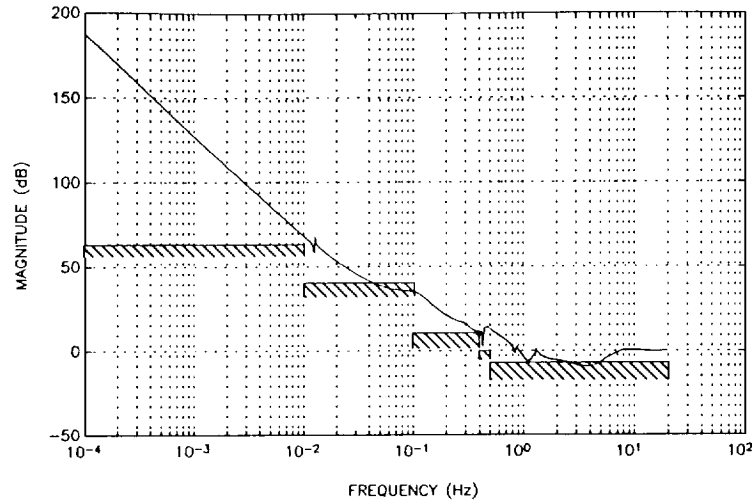


Figure 4.22 Return Difference Minimum Singular Value Frequency Response with Final Controller

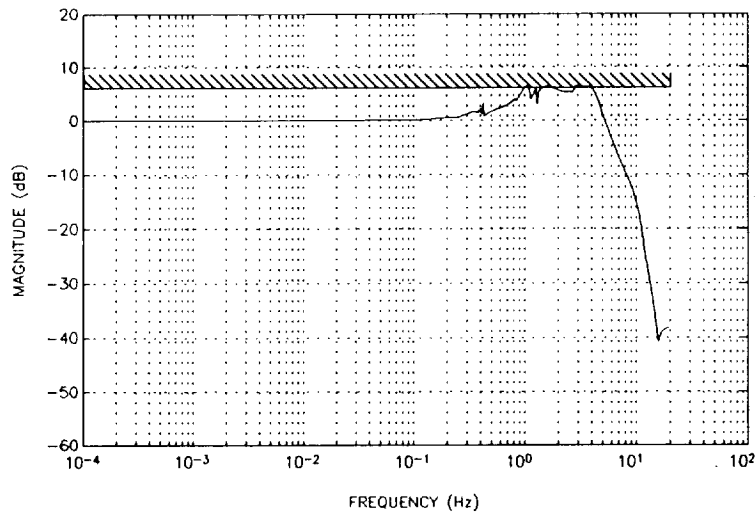


Figure 4.23 Complementary Sensitivity Maximum Singular Value Frequency Response with Final Controller

of the V_3 axis response shown in Figure 4.25 indicates a significant reduction in the peak near 1 Hz, as well as reductions at other frequencies.

Table 4.3 Peak and RMS Values for Linear Simulation with Final Controller

	HST Axis		
	V_1	V_2	V_3
Peak Value (milliarcsec)	105.1	17.03	36.59
RMS Value (milliarcsec)	28.60	2.969	7.239

During the course of this design effort several weaknesses of Algorithm A2 were discovered. The most important of these was the realization that the requirement that all the ordinary constraint violations and the worst violation of every frequency dependent constraint to decrease simultaneously was too restrictive. The algorithm continually failed to make significant progress before terminating, and it became necessary to constantly change the design constraints in order for the algorithm to proceed. Finally, reasonable performance and a successful design were obtained after altering the algorithm to accept the trial parameter vector if the worst violation of *all* the constraints decreased (the same test as used in the Polak-Mayne Algorithm). The new method for handling the ϵ -active constraints was retained, however.

It has also been observed that after running the algorithm many times with and without the in-the-large directions (singular value constraints only), that the in-the-large directions do not appear to have a significant impact on overall performance. Analysis of the method by which the search direction is calculated may offer one explanation. Although the in-the-large directions provide more information about constraint function behavior than gradients alone, the fact that the search direction is a compromise between many in-the-large directions and gradients probably eliminates the potential benefits of the in-the-large directions. Although this method of using the in-large-directions did not produce measurable improvements in algorithm performance, it is still felt that further research may lead to more effective ways to use this information, especially when the constraints enter into a condition of local opposition.

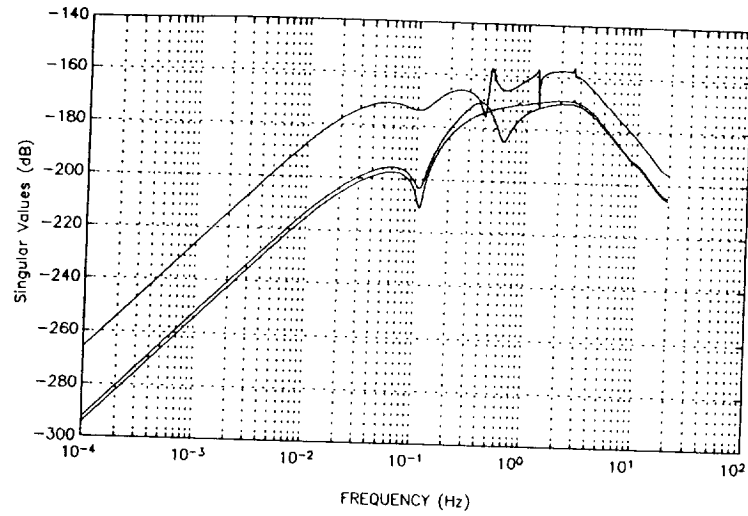


Figure 4.24 Singular Value Frequency Response from Disturbance to Attitude with Final Controller (All Three Singular Values Shown)

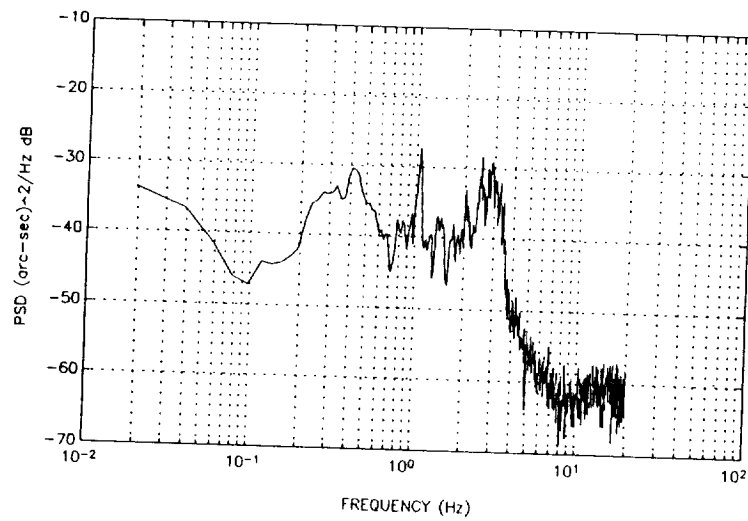


Figure 4.25 PSD Estimate of V_3 Axis Simulated Transient Response

4.9 A Performance Comparison

In the sub-section 4.8 two successful controller designs using Algorithm A1 and Algorithm A2 were presented. In this sub-section a study comparing the performance of Algorithm A3 and the Polak-Mayne Algorithm is presented. Direct performance comparisons of algorithms of this type can be difficult because of the many user-specified parameters that affect algorithm performance, and because while one algorithm may perform better than another on one problem, the reverse may occur on another problem. Therefore the results of a single comparison are not conclusive evidence that one algorithm is better than another. However, repeated experiments with several different problems can begin to lead to general conclusions about relative performance. The results of this particular design study are typical of those obtained from several experiments performed with these two algorithms.

4.9.1 The Design Problem

A design problem that is somewhat less complicated than the two problems that were presented in the previous sub-section 4.8 has been chosen to illustrate the relative performance of Algorithm A3 and the Polak-Mayne Algorithm. The plant is an arbitrary, stable, third order, two-input, two-output system taken from Polak and Mayne (1976). A sampling rate of 40 Hz was selected to convert the problem to discrete-time without significantly altering the frequency response over the bandwidth of the open-loop plant. The objective is to design a controller such that the steady-state error to a step input is zero, the closed-loop bandwidth is greater than 1 Hz, rejection of output disturbances exceeds 30 dB at all frequencies below 0.01 Hz, and stability robustness to output multiplicative unstructured uncertainty exceeds -1 dB at all frequencies below 2 Hz and -20 dB at 25 Hz. A block diagram of the closed-loop configuration is shown in Figure 4.26. The uncompensated open-loop singular value frequency response of the sampled-data system is shown in Figure 4.27.

The initial second-order controller was obtained by using LQG and model reduction techniques as described by Polak and Mayne. The singular value frequency response of the initial controller K is shown in Figure 4.28. The singular value frequency response of the transfer function from the reference r to the output y is shown in Figure 4.29 along with the corresponding closed-loop bandwidth specification. The singular value frequency response of the output sensitivity function is shown in Figure 4.30 along with the corresponding disturbance rejection specification. The singular value frequency response of the output complementary sensitivity function is shown in Figure 4.31 along with the corresponding stability robustness specification.

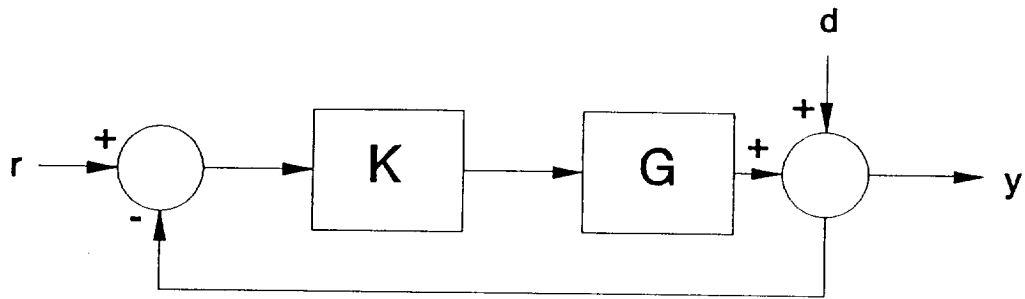


Figure 4.26 Block Diagram of the System

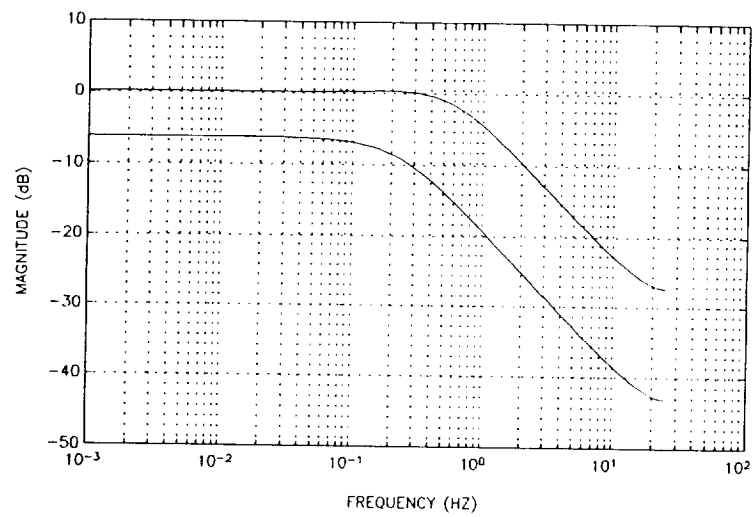


Figure 4.27 Open-Loop Singular Value Frequency Response

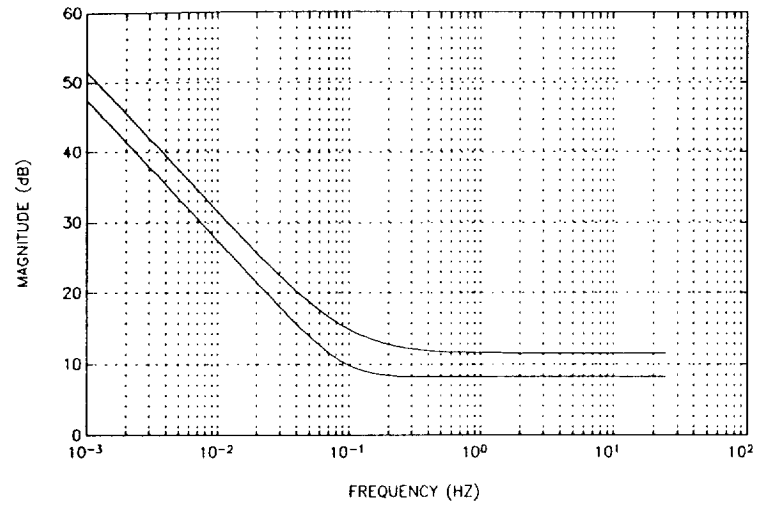


Figure 4.28 Initial Controller Singular Value Frequency Response

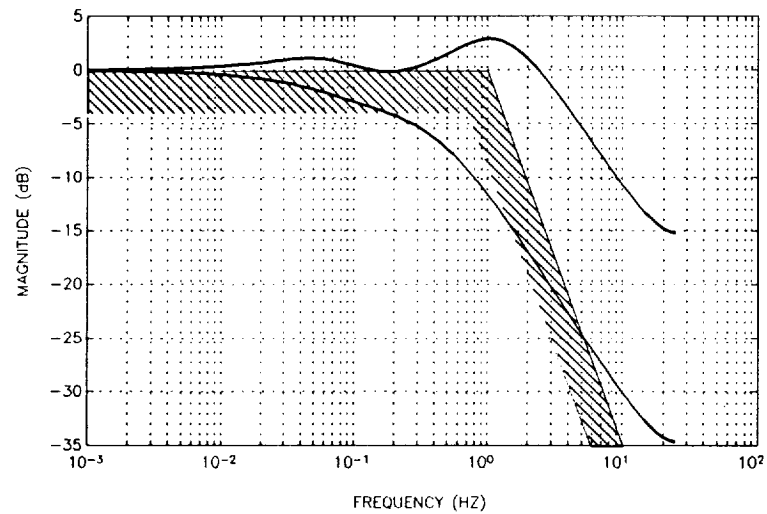


Figure 4.29 Initial Closed-Loop Transfer Function Singular Value Frequency Response and Specification

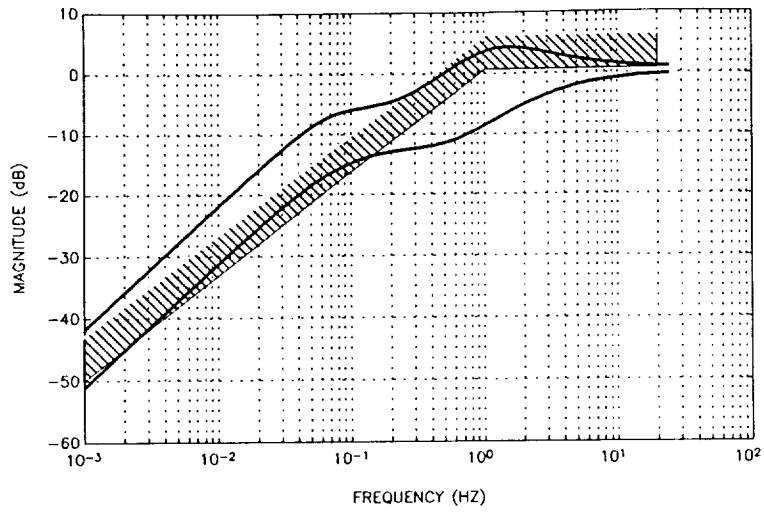


Figure 4.30 Initial Output Sensitivity Function Singular Value Frequency Response and Specification

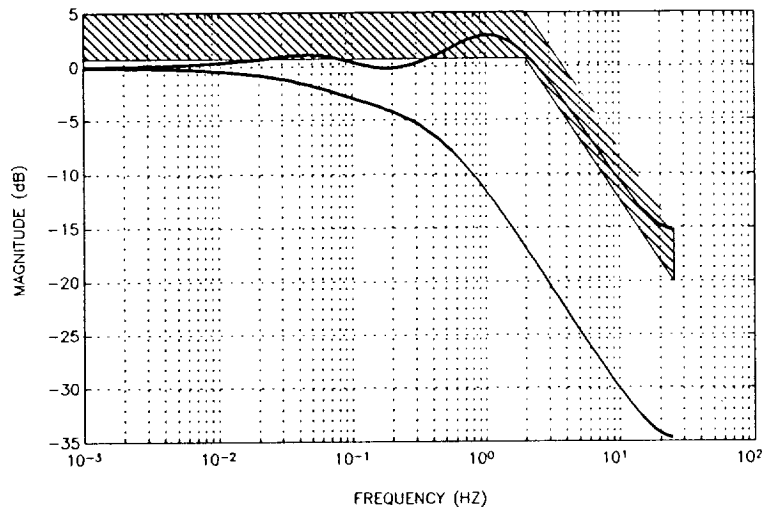


Figure 4.31 Initial Output Complementary Sensitivity Function Singular Value Frequency Response and Specification

4.9.2 Application of the Polak-Mayne Algorithm

The Polak-Mayne Algorithm was applied to the design problem using the initial controller described above with $\epsilon = 0.2$ and $\beta = 0.2$. The state feedback parameterization was used with the " F " matrix (sub-section 4.5.1) set equal to the zero matrix in order to achieve the zero steady-state error requirement. The algorithm required 523 iterations and 12 minutes, 22 seconds of CPU time on a Sun Sparcstation 2 to satisfy the design constraints. The singular value frequency response of the final controller, the singular value frequency response of the transfer function from the reference to the output, the singular value frequency response of the output sensitivity function, and the singular value frequency response of the output complementary sensitivity function are shown in Figures 4.32, 4.33, 4.34, and 4.35, respectively.

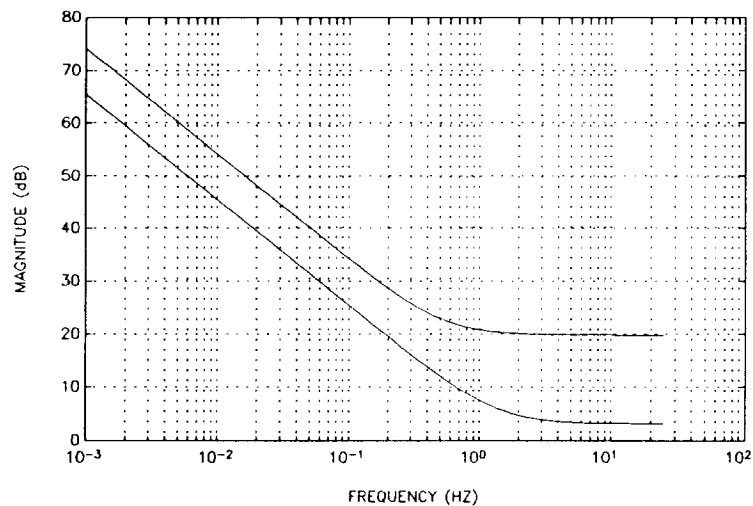


Figure 4.32 Controller Singular Value Frequency Response Designed by the Polak-Mayne Algorithm

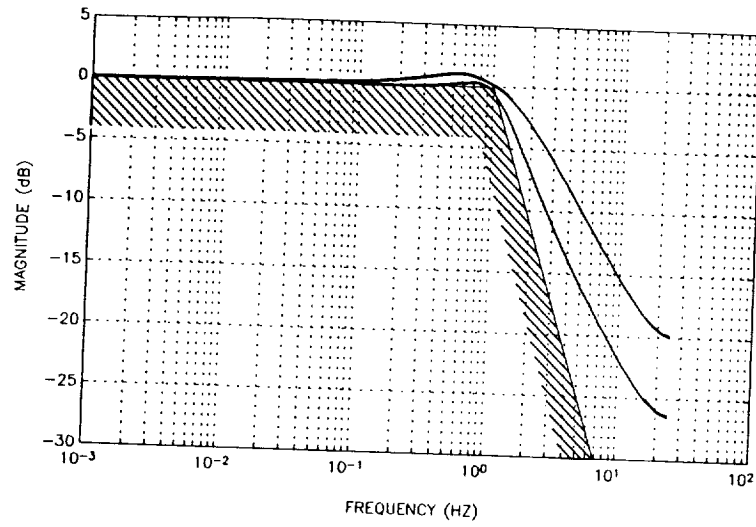


Figure 4.33 Closed-Loop Transfer Function Singular Value Frequency Response Designed by the Polak-Mayne Algorithm

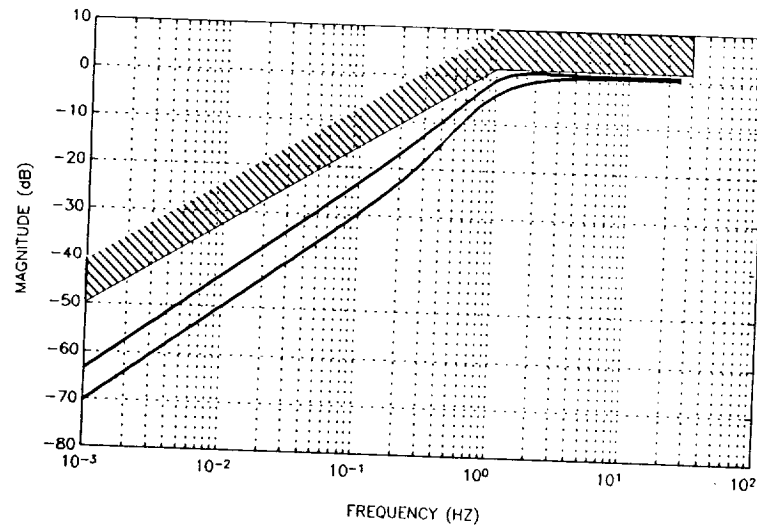


Figure 4.34 Output Sensitivity Function Singular Value Frequency Response Designed by the Polak-Mayne Algorithm

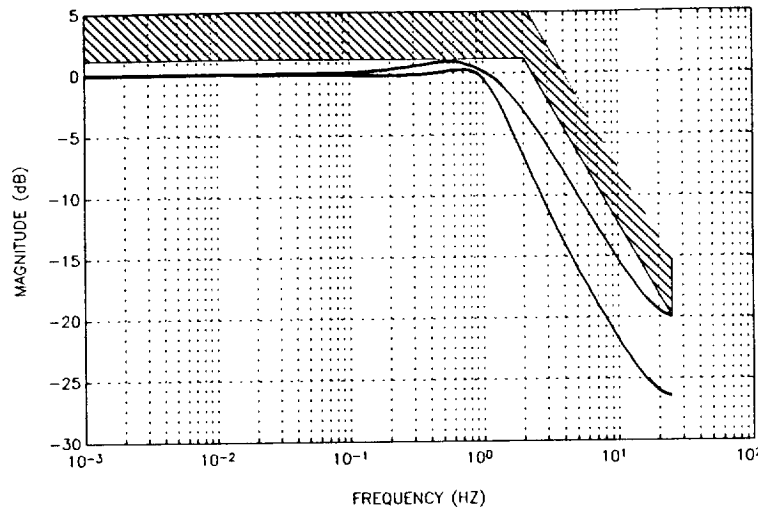


Figure 4.35 Output Complementary Sensitivity Function Singular Value Frequency Response Designed by the Polak-Mayne Algorithm

4.9.3 Application of Algorithm A3

Algorithm A3 was applied to the design problem using the initial controller described above with $\epsilon = 0.2$ and $\beta = 0.2$. The state feedback parameterization was used with the "F" matrix (sub-section 4.5.1) set equal to the zero matrix in order to achieve the zero steady-state error requirement. The algorithm required 251 iterations and 4 minutes, 33 seconds of CPU time on a Sun Sparcstation 2 to satisfy the design constraints. The singular value frequency response of the final controller, the singular value frequency response of the transfer function from the reference to the output, the singular value frequency response of the output sensitivity function, and the singular value frequency response of the output complementary sensitivity function are shown in Figures 4.36, 4.37, 4.38, and 4.39, respectively. Comparison of these results with those resulting from the Polak-Mayne Algorithm indicates that the final design in both cases was virtually the same.

The performance results indicate a significant improvement over the Polak-Mayne Algorithm in total iteration count and CPU time. It is important to reemphasize that the only difference between the two algorithms is the method used for calculating the search direction and that the search direction used by Algorithm A3 corresponds more closely to the measure of algorithm progress than that used by the Polak-Mayne Algorithm.

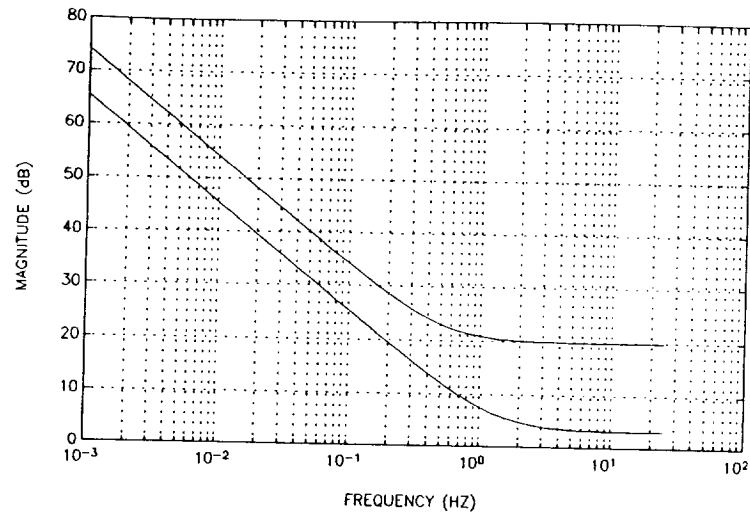


Figure 4.36 Controller Singular Value Frequency Response Designed by Algorithm A3

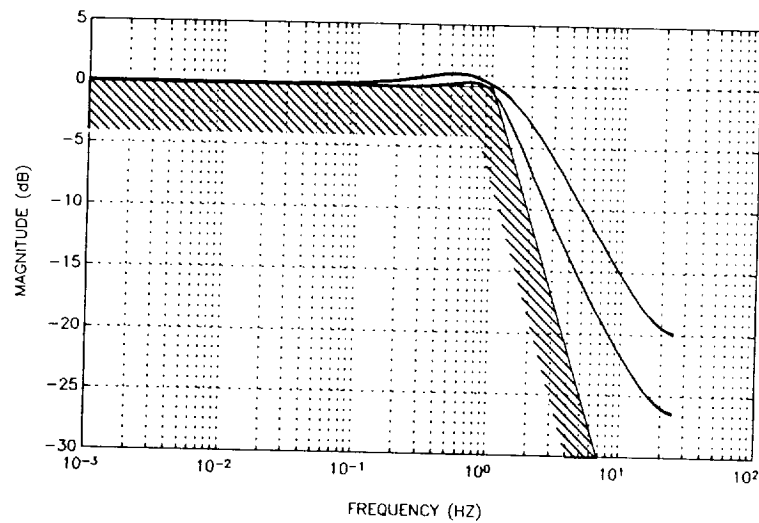


Figure 4.37 Closed-Loop Transfer Function Singular Value Frequency Response Designed by Algorithm A3

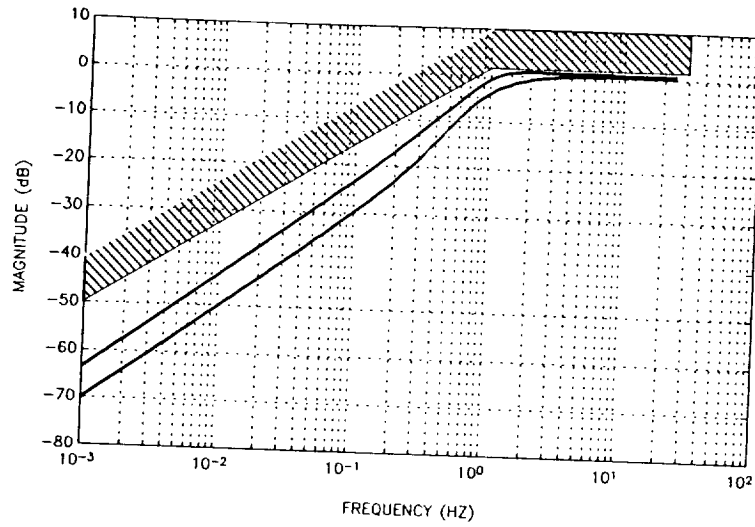


Figure 4.38 Output Sensitivity Function Singular Value Frequency Response Designed by Algorithm A3

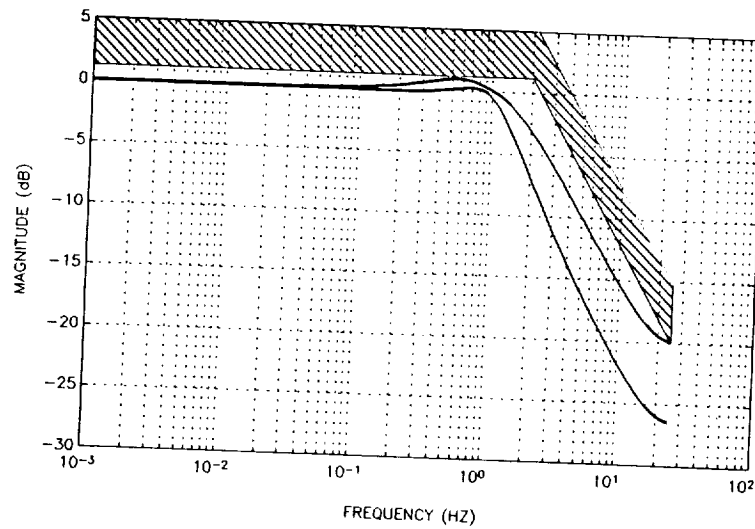


Figure 4.39 Output Complementary Sensitivity Function Singular Value Frequency Response Designed by Algorithm A3

4.10 Conclusions and Directions for Further Research for MADCADS

This research effort has involved the development and study of several mathematical programming algorithms that are suitable for search-based controller design. It has been illustrated that this approach to controller design has the advantages that it can simultaneously incorporate a wider variety and greater number of design constraints than modern analytical synthesis methods and can employ either parametric or nonparametric models of the plant as a basis for design. In fact, the application of a search-based method to controller design for a large aerospace structure ground test facility resulted in a controller that proved to be very effective upon actual implementation. Moreover, this design contained more constraints than could have been handled by any analytical synthesis method, and the design was performed without resorting to an analytical model of the plant.

Undoubtedly, the most significant weakness of search-based methods is the need for a good initial guess for the controller. As pointed out in sub-section 4.7, this is usually not a major difficulty if a parametric model of the plant is available, but it is a serious difficulty if the only model available is nonparametric. A systematic method of designing an initial controller from nonparametric data alone is certainly an area of research that needs to be pursued. The idea of using the Youla parameterization of all stabilizing controllers is a possible start in this direction. In fact, the difficulty of obtaining an initial controller points out that this approach to controller design is actually not a design methodology in itself, but a design improvement methodology. This does not lessen its utility however, since search-based methods can simultaneously incorporate far more constraints and a wider variety of constraints than any currently available analytical controller synthesis method.

Although the use of the in-the-large directions in Algorithm A2 did not produce any measurable performance improvements, it is felt that it was not the in-the-large directions themselves, but the method in which they were used in the search direction calculation that limited their value. However, it seems reasonable to believe that the more that is known about constraint function behavior, the better the algorithm will perform. This is certainly the case in unconstrained optimization. Therefore, the real issue for further research in this particular area is to discover how to effectively incorporate additional function information into the search direction calculation.

Another aspect of search-based methods for controller design improvement open to further study is the parameterization of the controller. As was illustrated in sub-section 4.5, controller parameterizations are not unique, which leads to the

question of which parameterization is best suited for design. The importance of this question was revealed when it was observed on several occasions that algorithm progress improved by simply performing a random change on the state coordinates of the state-space realization of the controller.

There are several contributions of this research effort to search-based methods for controller design. The most significant of these is the development of the method used for calculating the search direction in Algorithm A3. The key to the success of this algorithm is the fact that the search direction has a close correspondence to the measure of algorithm progress. As pointed out in sub-sections 4.4 and 4.9, the poor performance of the Polak-Mayne Algorithm can be attributed to this lack of correspondence.

A second contribution is reaffirmation of the difficulties that are encountered when a poor choice is made for a measure for algorithm progress, especially a measure that requires all violated constraints to improve simultaneously. As first encountered by CIT, and reen countered in Algorithm A1 and Algorithm A2, this measure appears to be too restrictive, although it is highly desirable from a theoretical viewpoint in the sense that the final value of each constraint can be no worse than the initial value of each constraint. It is possible, however, that the difficulty is not due entirely to the measure itself; but that the proper choice of search direction for the measure has not been discovered.

A third contribution is reinforcement of the value of using nonparametric plant models obtained directly from experimental data as the basis for design. This was accomplished through the implementation of a successful controller designed entirely from nonparametric plant models. Although the use of nonparametric plant models is not new, it has largely been ignored in modern control theory because of the inability of modern analytical synthesis techniques to use these models. However, with continued research in the area of search-based methods for controller design, the use of nonparametric plant models will undoubtedly increase.

Other contributions include the parameterizing of the controller by the state feedback parameterization and the associated method for fixing specified poles of the controller, the development of generalized expressions for the calculation of the partial derivatives of frequency response matrices with respect to controller parameters, and the efficient method for software implementation of the calculation of the partial derivatives of frequency dependent constraints through the use of a symbolic chain rule.

5 Conclusions

This section summarizes the results of Ohio University's efforts to advance the state-of-the-art of controller design using data models. Also included is a set of suggestions for future development.

5.1 Summary of Results

The major contributions of the grant effort have been the enhancement of the Compensator Improvement Program (CIP), which resulted in the Ohio University CIP (OUCIP) package, and the development of the Model and Data-Oriented Computer Aided Design System (MADCADS).

Incorporation of direct z -domain designs into CIP was tested and determined to be numerically ill-conditioned for the type of lightly damped problems for which the development was intended. Therefore, it was decided to pursue the development of z -plane designs in the w -plane, and to make this conversion transparent to the user. The analytical development needed for this feature, as well as that needed for including compensator damping ratios and DC gain specifications, closed loop stability requirements, and closed loop disturbance rejection specifications into OUCIP are all contained in Section 3. OUCIP was successfully tested with several example systems to verify proper operation of existing and new features.

The extension of the CIP philosophy and algorithmic approach to handle modern multivariable controller design criteria was implemented and tested. Several new algorithms for implementing the search approach to modern multivariable control system design were developed and tested. This analytical development, most of which was incorporated into the MADCADS software package, is described in Section 4, which also includes results of the application of MADCADS to the MSFC ACES facility and the Hubble Space Telescope.

5.2 Satisfaction of Objectives

As mentioned in the introduction to this document, only part of the proposed research for this grant was completed due to a lack of full funding. This sub-section reviews the original proposed research and development tasks and specifies which tasks were completed.

5.2.1 Completion of CIP Objectives

The following is a list of originally proposed tasks for the enhancements to CIP, with indications as to which were completed.

- (1) Perform modifications so either digital z-domain or continuous s-domain controllers could be produced. Task completed.
- (2) Provide the option for the inclusion of vibration suppression and disturbance/noise rejection specifications. Task completed.
- (3) Provide the option of independent frequency response specifications for each loop. Task not completed.
- (4) Include the option of specifying the controller in a state-space format. Task not completed.
- (5) Modify so that closed loop specifications could be made. Task completed.
- (6) For open loop specifications, provide the option of locating the loop breaking points either before the plant or before the controller. Task not completed.
- (7) Include pre and post-analysis of system singular values. This task was completed by the addition of pre and post-analysis graphical output.
- (8) Update the CIP code to operate in a workstation environment, i.e., user friendly and interactive, rather than a batch environment. Task completed. A modern graphical user interface was included in OUCIP.

5.2.2 Completion of Search-Based Modern Multivariable Control System Design Software Package Objectives

The following list contains the originally proposed items for the modern multivariable control system design software package and their status at the end of the grant.

- (1) Incorporation of a full complement of modern multivariable performance and robustness criteria into the existing code. Task completed.

- (2) The incorporation of damping ratio constraints. Task completed.
- (3) The incorporation of single input/output pair transmission constraints. Task completed.
- (4) The investigation and implementation of the most desirable state-space structures to realize a particular set of design constraints. Although a complete characterization of what controller structure is the most appropriate for a given set of design constraints was not found, several state-space controller structures were studied.
- (5) The implementation of the design software in a professional workstation environment. Task completed.
- (6) The development of effective graphical algorithm evaluation aids to allow effective designer interaction was completed. Graphical output of frequency responses and the possibility of stopping the iterative process, saving the solution and adjusting iteration parameters makes user interaction with the algorithm possible.

5.2.3 Completion of Software Testing and Application Objectives

This sub-section briefly addresses the testing and application issues regarding both CIP and MADCADS. The originally proposed tasks and their status at project termination are listed below.

- (1) The application of the enhanced CIP to the problem of vibration suppression for the Hubble Space Telescope (HST) solar power panels. This problem was not addressed using CIP but was solved using MADCADS.
- (2) The application of the modern multivariable search-based design algorithms to the HST vibration suppression problem. Task completed via MADCADS.
- (3) The application of the multivariable algorithms to the CASES ground facility. Task not completed.
- (4) The application of enhanced CIP to the Single Structure Control facility. Task not completed.

5.3 Future Work

The most important suggestion that can be made for future development is the consolidation of OUCIP and MADCADS into a single software package that includes all features of both programs and some features that have not yet been implemented in either software package. This combination would be synergistic in nature in that problems could then be addressed that would not be solvable with individual application of the design techniques.

Another important recommendation for future work is the completion of those tasks originally proposed that were not completed. These tasks include adding the option of independent frequency response specifications for each broken loop, providing the ability to break the loops either before the plant or before the forward path controller, and using the software in the design/fine-tuning of controllers for several test facilities.

In addition to combining the separate packages other enhancements could be included to greatly expand the capabilities of the software and create an even more user-friendly environment. Such enhancements could include

- (1) Graphically interactive block diagram(s), through which the user could manipulate the data corresponding to each block. Both frequency response and constraint data could be input graphically, through data files, or via parametric methods.
- (2) On-line search algorithm selection. In addition to gradient techniques the user would be able to choose stochastic methods such as simulated annealing.
- (3) Additional types of design constraints and analysis tools. This would allow the user to tackle a broader range of problems and facilitate system analysis.

OUCIP and MADCADS are professional quality software packages. The completion of the above suggested enhancements would yield a design package that will be able to handle most control system design problems.

6 References

- Anderson, Brian D.O. and Moore, J.B. (1990), *Optimal Control: Linear Quadratic Methods*, Prentice-Hall, Englewood Cliffs, New Jersey
- Boyd, S.P., Balakrishnan, V., Barrat, C.H., et. al. (1988), "A New CAD Method and Associated Architectures for Linear Controllers", *IEEE Transactions on Automatic Control*, vol. AC-23, pp. 268-283.
- Churchill, R.V. and Brown, J.W. (1990), *Complex Variables and Applications*, 5th ed., McGraw-Hill, New York
- Davis, L., Collins, E., and Hodel, A.S. (1992), "A Parameterization of Minimal Plants," *Proceedings of the 1992 American Control Conference*, pp. 355,356
- Dongarra, J.J., Moler, C.B., Bunch, J.R., and Stewart, G.W. (1979), *LINPACK Users' Guide*, Society for Industrial and Applied Mathematics, Philadelphia
- Duncan, M.A. (1994), "Enhancements to the Compensator Improvement Program", Master of Science Thesis, Ohio University, Athens, OH, March, 1994.
- Edmunds, J.M. (1979), "Control System Design and Analysis Using Closed-loop Nyquist and Bode Arrays," *International Journal of Control*, vol. 30, pp. 773-802
- Feldman, S.I., Gay, D.M., Maimone, M.W. and Schryer, N.L. (1993), "A Fortran-to-C Converter", Computing Science Technical Report No. 149, AT&T Bell Laboratories, Murray Hill, New Jersey, March, 1993.
- Francis, B.A. (1987), *A Course in H_∞ Control Theory*, Lecture Notes in Control and Information Sciences, vol. 88, Springer-Verlag, Berlin
- Frazier, W.G. and Irwin, R.D. (1991), "Numerical Techniques for Simultaneous Achievement of Multiple Design Constraints for Multivariable Control Systems", *Proceedings of the Twenty-Third Southeastern Symposium on System Theory*, University of South Carolina, Columbia, SC, March 1991.
- Frazier, W.G. and Irwin, R.D. (1993), "Designing Reduced-order Multivariable Controllers Using Experimentally Derived Plant Data," *Journal of Guidance, Control, and Dynamics*, vol. 16, No. 1, pp. 53-58

- Frazier, W.G. (1993), "Search-Based Methods for Computer-Aided Controller Design Improvement", Ph.D. Dissertation, Ohio University, Athens, OH, June, 1993.
- Garbow, B.S., Boyle, J.M., Dongarra, J.J., and Moler, C.B. (1977), *EISPACK Guide Extension*, Springer-Verlag, New York
- Gill, Philip, E., Murray, Walter, and Wright, Margaret H. (1981), *Practical Optimization*, Academic Press, New York
- Glassner, Andrew S., editor, (1990), *Graphics Gems*, Academic Press, Cambridge
- Glover, K. and Doyle, J.C. (1988), "State-space Formulae for All Stabilizing Controllers that Satisfy and H^∞ Norm Bound and Relations to Risk Sensitivity", *Systems and Controls Letters*, vol. 11, pp. 167-172.
- Golub, Gene H. and Van Loan, Charles F. (1989), *Matrix Computations*, 2 ed., The Johns Hopkins University Press, Baltimore
- Haddad, W.M., Bernstein, D.S., and Mustafa, D. (1991), "Mixed-Norm H^2/H^∞ Regulation and Estimation: The Discrete-Time Case", *Systems and Control Letters*, vol. 16, pp. 235-248.
- Heller, Dan, *Motif Programming Manual*, O'Reilly & Associates, Inc., Sebastopol, CA, 1990.
- Irwin, R. Dennis and Mitchell, Jerrel R. (1991), "Controller Design of a Large Space Structure Test Facility Using Experimental Data", *Proceedings of the Twenty-Third Southeastern Symposium on System Theory*, University of South Carolina, Columbia, SC, March 1991.
- Irwin, R.D., Glenn, R.D., Frazier, W.G., Lawrence, D.A., and Follet, R.F. (1995), "Analytically and Numerically Derived H^∞ Controller Designs for Hubble Space Telescope," *Journal of Guidance, Control, and Dynamics*, vol. 18, No. 2, pp. 214-221.
- Junkins, John and Kim, Youdan (1990), "First and Second Order Sensitivity of the Singular Value Decomposition", *The Journal of the American Astronautical Society*, Jan.-Mar.

- Kreisselmeier, G. and Steinhauser, R. (1979), "Systematic Control Design by Optimizing a Vector Performance Index," *Proceedings of the IFAC Symposium on Computer-Aided Design of Control Systems*, Zurich, pp. 113-117
- Kailath, Thomas (1980), *Linear Systems*, Prentice-Hall, Englewood Cliffs, New Jersey
- Kreyszig, E. (1988), *Advanced Engineering Mathematics*, Sixth Edition, John Wiley & Sons, Inc., New York, NY.
- Lawson, C.L. and Hanson, R. J. (1974), *Solving Least-Squares Problems*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey
- Luenberger, D.G. (1984), *Linear and Nonlinear Programming*, Addison-Wesley, Reading, Mass.
- Maciejowski, J.M. (1989), *Multivariable Feedback Design*, Addison-Wesley, Reading, Mass.
- Mitchell, Jerrel R. (1972), *An Innovative Approach to Compensator Design*, Ph.D. dissertation, Mississippi State University, State College, Miss.
- Mitchell, Jerrel R. (1973), "An Innovative Approach to Compensator Design", NASA Contractor Report, CR-2248, May 1973.
- Mitchell, Jerrel R., McDaniel, W. L. Jr., and Gresham L. L. (1977), "Compensator Improvement for Multivariable Control Systems", Final Report, Contract No. NAS8-31568, NASA/MSFC, August 1977.
- Mitchell, Jerrel R., McDaniel, W. L. Jr., and Gresham L. L. (1980), "A Multivariable Control System Design Algorithm", *AIAA Journal of Guidance and Control*, Vol. 3, No. 4, July-August 1980.
- Mitchell, Jerrel R. (1984), "1-CAT (One Controller at a Time): A Frequency Domain Multi-Input, Multi-Output Design Approach", *Proceedings of the 1984 AIAA Guidance and Control Conference*, Seattle, WA, August 1984.
- Newsom, Jerry R. and Mukhopadhyay, V. (1985), "A Multiloop Robust Controller Design Study Using Singular Value Gradients," *The Journal of Guidance, Control and Dynamics*, vol. 8, pp. 514-519

- Ogata, K. (1990), *Modern Control Engineering*, Second Edition, Prentice Hall, Inc., Englewood Cliffs, NJ.
- Polak, Elijah and Mayne, David Q. (1976), "An Algorithm for Optimization Problems with Functional Inequality Constraints," *IEEE Transactions on Automatic Control*, vol. AC-21, no. 2
- Postlethwaite, I., MacFarlane, A.G.J., and Edmunds, J.M. (1981), "Principle Gains and Principal Phases in the Analysis of Linear Multivariable Feedback Systems," *IEEE Transactions on Automatic Control*, vol. AC-26, no. 2, pp. 32-46
- Sharkey, J.P, Nurre, G.S., Beals, G.A., and Nelson, J.D., "A Chronology of the On-Orbit Pointing Control System Changes on the Hubble Space Telescope and Associated Pointing Improvements," *AIAA Guidance and Control Conference*, 1992
- Smith, B.T., Boyle, J.M., Dongarra, J.J., Garbow, B.S., Ikebe, Y., Klema, V.C., and Moler, C.B. (1976), *EISPACK Guide*, 2 ed., Springer-Verlag, New York
- Wie, B., Liu, Q., and Bauer, F. (1993), "Classical and Robust H_{∞} Control Redesign for the Hubble Space Telescope", *The Journal of Guidance, Control and Dynamics*, vol. 16, No. 6, pp. 1069-1077
- Youla, D.C., Jabr, H.A., and Bongiorno, J.J.Jr. (1976), "Modern Wiener-Hopf Design of Optimal Controllers - Part II: The Multivariable Case," *IEEE Transactions on Automatic Control*, vol. AC-21, no. 3, pp. 319-338
- Zakian, V. and Al-Naib, U. (1973), "Design of Dynamical and Control Systems by the Method of Inequalities," *Proceedings of the Institution of Electrical Engineers*, vol. 120, pp. 1421-1427

Appendix A: Multivariable Taylor Series

The following discussion of the multivariable Taylor series is based upon the presentation of Gill, Murray, and Wright (1981).

Let $x \in R^n$, $p \in R^n$ such that $\|p\|_2 = 1$, and let $h \in R$ be positive. Let $f: R^n \rightarrow R$ be a function such that all partial derivatives of order 2 exist and are continuous. Then there exists $\theta \in [0,1]$ such that

$$f(x + hp) = f(x) + hDf(x)p + \frac{1}{2}h^2p^TD^2f(x + h\theta p)p \quad (A.1)$$

where

$$Df = \left[\frac{\partial f}{\partial x_1} \quad \frac{\partial f}{\partial x_2} \quad \dots \quad \frac{\partial f}{\partial x_n} \right]^T \quad (A.2)$$

is the gradient of f and D^2f is the Hessian, a matrix whose (i,j) entry is given by $(D^2f)_{ij} = \frac{\partial^2 f}{\partial x_i \partial x_j}$. The value of being able to represent a function in the form of Equation A.1 is that it allows for the approximation of the function in the neighborhood of some point x by less complicated linear and quadratic functions. (Note: In general a Taylor series may consist of the partial derivatives of many orders. The restriction to second order partials has been in order to simplify notation.)

One of the most important implications of the Taylor series for mathematical programming algorithms is the following result. Suppose f has the Taylor series expansion

$$f(x + hp) = f(x) + hDf(x)p + \frac{1}{2}h^2p^TD^2f(x + h\theta p)p \quad (A.3)$$

where $\|Df(x)\| \neq 0$. Then there exist p and h such that $f(x + hp) > f(x)$. To prove this result let p be such that $Df(x)p > 0$. Such a p always exists since p can be chosen to be $Df(x)$. Since the entries of D^2f are continuous at x , the quantity $p^TG(x + h\theta p)p$ is bounded on some interval $h \in [0, h_0]$, $h_0 > 0$. Let $m = \min_{h \in [0, h_0]} p^TG(x + h\theta p)p$. Then

$$f(x + hp) - f(x) \geq h \left[Df(x)p + \frac{h}{2} \frac{m}{1 + |m|} \right], \quad h \in [0, h_0]. \quad (A.4)$$

If h is then chosen such that $0 < h < \min \left\{ h_0, \frac{2Df(x)p}{1 + |m|} \right\}$ then it follows that

$$f(x + hp) - f(x) \geq 0. \quad (A.5)$$

Q.E.D.

Appendix B: Cauchy-Riemann Equations

The following presentation of the Cauchy-Riemann equations is taken from Churchill and Brown (1990). Let $f: C \rightarrow C$ be an analytic function of a complex variable $z = (x + jy)$ as defined by

$$f(z) = u(x,y) + jv(x,y) , \quad (\text{B.1})$$

where $u: R^2 \rightarrow R$ and $v: R^2 \rightarrow R$ are functions for which all first-order partial derivatives exist. Then the following equations, commonly referred to as the Cauchy-Riemann equations, hold:

$$\text{and} \quad \frac{\partial u}{\partial x}(x,y) = \frac{\partial v}{\partial y}(x,y) \quad (\text{B.2})$$

$$\frac{\partial u}{\partial y}(x,y) = -\frac{\partial v}{\partial x}(x,y) . \quad (\text{B.3})$$

Moreover, it can also be shown that

$$\frac{df}{dz}(z) = \frac{\partial u}{\partial x}(x,y) + j \frac{\partial v}{\partial y}(x,y) . \quad (\text{B.4})$$

Appendix C: Nomenclature

\mathcal{C}	the set of complex numbers
\mathcal{C}^n	the set of complex-valued vectors of dimension n
$\mathcal{C}^{m \times n}$	the set of complex-valued matrices with m rows and n columns
$Im\{z\}$	the imaginary part of the complex number z
\mathcal{R}^n	the set of real-valued vectors of dimension n
$\mathcal{R}^{m \times n}$	the set of real-valued matrices with m rows and n columns
$Re\{z\}$	the real part of the complex number z
$\sigma_i(A)$	the i th largest singular value of the matrix A
$\sigma_{\max}(A)$	the maximum singular value of the matrix A
$tr(A)$	the trace of the square matrix A
A^H	the complex conjugate transpose of the matrix A
A^T	the transpose of the matrix A
\in	'an element of'
\forall	'for every'

Appendix D: Brief Description of the Software Packages

The development documented in this report has produced two computer applications: OUCIP and MADCADS. At the time of this writing, both programs have been compiled for the UNIX operating system versions SunOS4.1.3 and DELL UNIX System V Release 4.0.

For SunOS4.1.3, C modules were compiled with gcc, version 2.5.8, and FORTRAN modules were compiled with f77, the Sun FORTRAN compiler, version 1.4. Objects modules were linked with f77.

For DELL UNIX, which is the operating system used by ED12B at NASA Marshall Space Flight Center, FORTRAN modules were compiled by invoking the script f77. This script uses the program f2c (Feldman, 1993) to convert FORTRAN modules to the C language and then compiles the resulting C modules with the installed C compiler cc. Linking was performed using cc.

In both OUCIP and MADCADS, the numerical computations are performed in FORTRAN, and the widely available packages EISPACK (Garbow, 1977), LINPACK (Dongarra, 1979), and/or BLAS (Dongarra, 1979) are used for basic linear algebra computations and eigensystem solutions, as appropriate.

The graphical user interfaces (GUIs) for both programs were written in C working with the X Window System, release X11R5, using the Motif Toolkit, release 1.2.4. Some of the GUI modules are modified versions of the C programs included in (Heller, 1991). The following is the Copyright notice that accompanies these programs.

```
/* Written by Dan Heller. Copyright 1991, O'Reilly && Associates.  
 * This program is freely distributable without licensing fees and  
 * is provided without guarantee or warranty expressed or implied.  
 * This program is -not- in the public domain.*/
```

Loose labels for plots were implemented by using a C routine written by Paul Heckbert, which can be found in (Glassner, 1990). The following note accompanies that software.

The authors and the publisher hold no copyright restrictions on any of these files; this source code is public domain, and is freely available to the entire computer graphics community for study, use, and modification. We do request that the comment at the top of each file, identifying the original author and its original publication in the book Graphics Gems, be retained in all programs that use these files.

The matrix widget XbaeMatrix, version 3.8, was used in some dialog boxes. The use of matrices provides ease of data input in these cases. The Copyright status of the XbaeMatrix package is given by the following note. The release of XbaeMatrix used included third party modifications, as written at the end of the note.

```
/*
 * Copyright(c) 1992 Bell Communications Research, Inc. (Bellcore)
 * All rights reserved
 * Permission to use, copy, modify and distribute this material for any purpose and without fee
 * is hereby granted, provided that the above copyright notice and this permission notice appear
 * in all copies, and that the name of Bellcore not be used in advertising or publicity pertaining to
 * this material without the specific, prior written permission of an authorized representative of
 * Bellcore.
 *
 * BELLCORE MAKES NO REPRESENTATIONS AND EXTENDS NO WARRANTIES, EXPRESS OR
 * IMPLIED, WITH RESPECT TO THE SOFTWARE, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR ANY PARTICULAR
 * PURPOSE, AND THE WARRANTY AGAINST INFRINGEMENT OF PATENTS OR OTHER
 * INTELLECTUAL PROPERTY RIGHTS. THE SOFTWARE IS PROVIDED "AS IS", AND IN NO
 * EVENT SHALL BELLCORE OR ANY OF ITS AFFILIATES BE LIABLE FOR ANY DAMAGES,
 * INCLUDING ANY LOST PROFITS OR OTHER INCIDENTAL OR CONSEQUENTIAL DAMAGES
 * RELATING TO THE SOFTWARE.
 *
 * MatrixWidget Author: Andrew Wason, Bellcore, aw@bae.bellcore.com
 *
 * Modification History: David Boerschlein, dpb@air16.larc.nasa.gov
 * Lockheed Engineering & Sciences Company,
 * Under contract to NASA LaRC:
```

The Xvertex set of routines was used for the drawing of rotated labels. The following notice describes the status of Xvertex.

```
/* ...../
/* xvertex 5.0, Copyright (c) 1993 Alan Richardson (mppa3@uk.ac.sussex.syma)
 *
 * Permission to use, copy, modify, and distribute this software and its
 * documentation for any purpose and without fee is hereby granted, provided
 * that the above copyright notice appear in all copies and that both the
 * copyright notice and this permission notice appear in supporting
 * documentation. All work developed as a consequence of the use of
 * this program should duly acknowledge such use. No representations are
 * made about the suitability of this software for any purpose. It is
 * provided "as is" without express or implied warranty.
 */
/* ...../
```

Appendix E: OUCIP User's Guide

OUCIP:
Ohio University
Compensator Improvement Program

User's Guide

Enrique A. Medina

Jerrel R. Mitchell

Department of Electrical and Computer Engineering
Ohio University
Athens, Ohio 45701

July 7, 1995
Department of Electrical and Computer Engineering
Ohio University
Stocker Engineering Center
Athens, Ohio 45701

This work was supported in part by a grant from NASA Marshall Space Flight Center

SunOS and OpenWindows are trademarks of Sun Microsystems, Inc.
UNIX is a registered trademark of UNIX System Laboratories, Inc.
X Window System is a product of the Massachusetts Institute of Technology.
Motif is a registered trademark of the Open Software Foundation.
Postscript is a registered trademark of Adobe Systems, Inc.

Table of Contents

1.0 Introduction	E4
1.1 System Description, Design Goals and Design Process	E4
1.2 Getting Started	E6
1.3 Organization of this Guide	E8
2.0 Data Input	E8
3.0 Graphical User Interface and Data Files Formats	E10
3.1 <i>FILE</i> Menu	E10
3.1.1 <i>Settings</i> E10; 3.1.2 <i>Plant</i> E12; 3.1.2 <i>Compensator</i> E13; 3.1.4 <i>Specifications</i> E14; 3.1.5 <i>Application</i> E16; 3.1.6 <i>Text Logs</i> E16; 3.1.8 <i>Quit</i> E17	
3.2 <i>PARAMETERS</i> Menu	E18
3.2.1 <i>Title</i> E18; 3.2.2 <i>Mode</i> E18; 3.2.3 <i>Step Size</i> E18; 3.2.4 <i>Maximum Step Size</i> E18; 3.2.5 <i>Minimum Step Size</i> E18; 3.2.6 <i>Hysteresis Threshold</i> E19	
3.3 <i>EXECUTE</i> Menu	E19
3.3.1 <i>Single</i> E19; 3.3.2 <i>Dialog</i> E19; 3.3.3 <i>Pre/Post Analysis</i> E20	
3.4 <i>GRAPHICS</i> Menu	E20
3.4.1 <i>Create Window</i> E20; 3.4.2 <i>Kill Window</i> E22; 3.4.3 <i>Window Size</i> E22; 3.4.4 <i>Clear Windows</i> E22; 3.4.5 <i>Kill All Windows</i> E22	
3.5 <i>SPECIFICATIONS</i> Menu	E22
3.5.1 <i>Relative Stability</i> E22; 3.5.2 <i>Disturbance Rejection</i> E23; 3.5.3 <i>Compensator DC Gain</i> E23; 3.5.4 <i>Compensator Zeroes Damping Ratios</i> E23; 3.5.5 <i>Compensator Poles Damping Ratios</i> E23	
3.6 <i>ACTIVATE</i> Menu	E24
3.6.1 <i>Relative Stability</i> E24; 3.6.2 <i>Disturbance Rejection</i> E24; 3.6.3 <i>Compensator DC Gain</i> E24; 3.6.4 <i>Compensator Damping Ratios</i> E24	
3.7 <i>HELP</i> Menu	E24
3.8 <i>OUCIP</i> Plots	E24
3.8.1 <i>Semilog Plots</i> E24; 3.8.2 <i>Polar Plots (and data sweep)</i> E25	
4.0 Example	E26
5.0 Conclusions	E33
6.0 References	E34
Appendix	E35

OUCIP: ***Ohio University*** ***Compensator Improvement Program*** ***User's Guide***

1.0 Introduction

OUCIP is a graphically-oriented, interactive piece of software that assists in the process of compensator design for multivariable dynamic systems by applying a search algorithm to vary the parameters of an initial stabilizing compensator in order to improve the values of performance measurements. The original Compensator Improvement Program (CIP) was developed in the 1970's at Mississippi State University (Mitchell, 1973). The term *OUCIP* refers to Ohio University's CIP, which is the most recent version, whose use is described in this document. A more detailed explanation of the underlying theory of *OUCIP* is given by Mitchell, et. al. [5]. At the time of this writing, *OUCIP* has been compiled and tested under SunOS4.1.3 and DELL UNIX System V Release 4.0. The graphical user interface (GUI) was written for the X Window System, Release 5, using the Motif™ Toolkit, version 1.2.4.

1.1 System Description, Design Goals and Design Process

The block diagram for a multi-input, multi-output, linear, time-invariant, discrete-time feedback control system is shown in Fig. 1, where $R \in \mathbb{R}^r$ is a vector of reference inputs, $X \in \mathbb{R}^m$ is a vector of plant control inputs, $D \in \mathbb{R}^d$ is a vector of disturbance inputs, $Y \in \mathbb{R}^p$ is a vector of measured outputs used for feedback, and $Z \in \mathbb{R}^{nz}$ is a vector of physical outputs that are not or cannot be used for feedback.

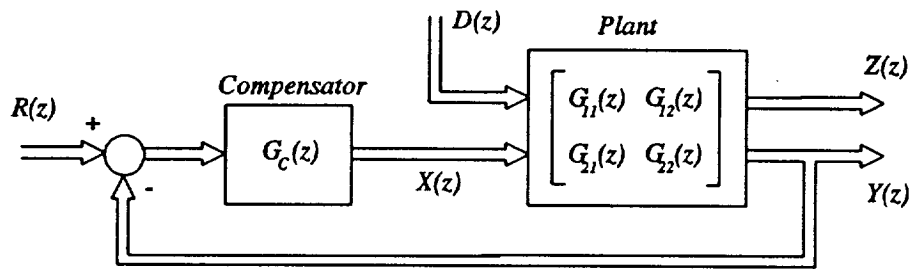


Figure 1 CIP Block Diagram

The plant transfer function matrix $G(z)$ is partitioned into blocks $G_{11}(z) \in \mathbb{R}^{nz \times d}$, $G_{12}(z) \in \mathbb{R}^{nz \times m}$, $G_{21}(z) \in \mathbb{R}^{p \times d}$, and $G_{22}(z) \in \mathbb{R}^{p \times m}$, which are the transfer function matrices from D to Z , X to Z , D to Y , and X to Y , respectively. (Throughout the

document, z-domain designs are assumed unless otherwise stated, although *OUCIP* can achieve designs in the s , z , or w planes.) The block $G_c(z) \in \mathbb{R}^{m \times p}$ is the compensator transfer function matrix. *OUCIP* is designed to vary the coefficients of the elements of $G_c(z)$ so that simultaneous improvements in one or more of the following types and/or combinations of design specifications are achieved: (1) relative stability, (2) modal attenuation (gain stabilization), (3) steady-state accuracy, (4) compensator robustness and (5) disturbance rejection (peak and rms values).

Relative stability performance measurements are obtained by viewing the system as m coupled feedback systems. *OUCIP* allows for the improvement of gain, phase, and stability margins and attenuation levels in each of the m open loop scalar frequency responses that are obtained when each feedback loop is opened between the compensator and the plant while the other loops remain closed. (This is called the broken loop method.) Desired closed loop performance measurements are obtained from peak and rms values of the frequency responses from $D(z)$ to $Y(z)$. Steady-state accuracy and robustness performances are measured by compensator DC gain values and damping ratios. The user can choose design specifications for any or all of these. *OUCIP* does not implement a loop-at-a-time procedure; the compensator is iteratively incremented so that all unsatisfied performance measures are improved or, as a minimum, not allowed to degrade.

The iterative process in *OUCIP* is controlled by the user. Each iteration can be outlined as follows:

- 1) Compute each broken loop frequency response and evaluate gain, phase and stability margins and attenuation levels for each.
- 2) Evaluate DC gains and damping ratios of each element of the compensator transfer function matrix.
- 3) Compute closed loop frequency responses and evaluate RMS and peak values of each element of the closed loop frequency response matrix from D to Y .
- 4) Determine which of the measurements evaluated in 1, 2, and 3 do not satisfy the design objectives. If all objectives are satisfied, notify and return control to the user.
- 5) Otherwise, with respect to the coefficients of the compensator transfer function matrix, calculate the gradient vectors of the performance measurements that do not comply with the design objectives.
- 6) Using these gradient vectors, compute a coefficient change vector that will assure the possibility of simultaneously improving one or more performance measurements while the others do not get worse. If no design objective can be significantly improved, notify and return control to the user.
- 7) Otherwise, using the change vector, increment the coefficients of the compensator transfer function matrix. Return control to the user or go back to step 1.

1.2 Getting Started

OUCIP is executed by typing *oucip* at the command prompt. The main window shown in Figure 2 will be drawn on the screen. There are three main elements in this window: the menu bar, the message area, and the scroll bars. The menu bar allows the user to select actions to carry out. These actions will be explained later in this document. *OUCIP* will send important execution and error messages to the message area. Information that has scrolled out of the visible portion of the message area can be seen by using the scroll bars. *OUCIP* will output information on the results of each iteration of the algorithm to the window from which it was executed or to a console window if the program is started by other means¹.

It is assumed in this guide that the user has basic knowledge about how to interact with the particular window manager being run and with software applications that use the Motif *look and feel*. Information on how to use the window manager should be available in the form of manuals that commonly accompany a workstation. A good source of information about how to use the Motif window manager and applications that run with Motif is the Motif User's Guide [6].

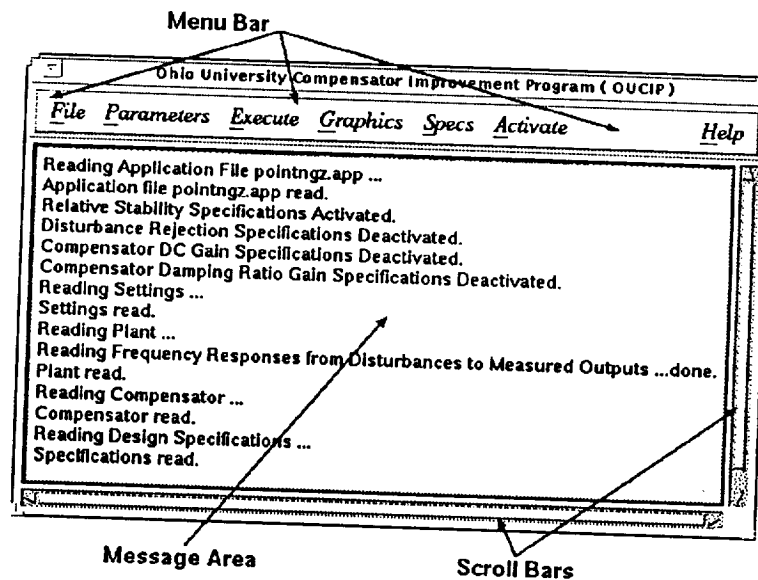


Figure 2 Oucip Main Window

In order to use *OUCIP* to improve a controller design, the user must first input the necessary data to define the problem. This is done by means of the *File* menu. Then the *Execute* menu is used to start and control the iterative process. The rest of this guide explains how to create data files, modify specifications, execute iterations, and

¹ In Openwindows™, for example, a program can be started by double-clicking on its icon in the file manager.

obtain text and graphical results. For the benefit of those users who want to gain some experience in using *OUCIP* before embarking into the tasks of solving problems of their own, several demonstration examples are included with the program. The following instructions briefly describe how to load a demo problem into *OUCIP* and start the iterative process. Once a user has created the appropriate files corresponding to a problem of interest, the process is the same.

- (1) Click¹ on the *File* item of the menu bar. This will create a "pulldown" menu with several options.
- (2) Click on the *Application* option of the pulldown menu. This will create a "cascade" menu with two options to the side of the pulldown menu.
- (3) Click on the *Retrieve...* option of the cascade menu. This will create a file selection box as shown in Figure 3.
- (4) Double-click² on the line that reads *pointngz.app* in the file list located on the right hand side of the file selection box. *OUCIP* will load the data for the z-domain pointing system example presented later in this guide. The file selection dialog should disappear.
- (5) Click on the *Execute* item of the menu bar. This will create a pulldown menu with three options.
- (6) Click on the *Single* option of the pulldown menu. This will run iteration 0 of the algorithm and output results to the window from which the program was started.
- (7) Click on the *Graphics* item of the menu bar and then on the *Create Window ...* option of the pulldown menu that is created. This will create a *Frequency Response Plot Definition* dialog box as shown in Figure 5.
- (8) Click on the *Default Title* button of the dialog box just created. Then click on the *Create* button of the same dialog box. This will create a window with a semi-logarithmic plot of the magnitude frequency response of the first broken loop.
- (9) At this point, other specifications can be activated, new plots can be created, and/or additional iterations can be run. The *Quit* option under the *File* menu will terminate the execution of *OUCIP*.

¹ "Click" means to press and release a mouse button without moving the pointer. Unless otherwise noted, this refers to the left mouse button in this guide.

² "Double-click" means to click a button twice in rapid succession. Unless otherwise noted, this refers to the left mouse button in this guide.

1.3 Organization of this Guide

The remainder of this document is divided into four sections. Section 2 describes the data needed in order to use *OUCIP* to improve a controller design. Section 3 is a reference of the graphical user interface, its various options, and the format used in input files. In Section 4 an example problem is presented. Some concluding remarks are given in Section 5. Finally, Section 6 lists several other sources of information regarding *OUCIP*.

2.0 Data Input

Before the search algorithm in *OUCIP* can be used to improve a control system design, the user must create and input four sets of data. These are the settings, plant frequency response, initial compensator, and specifications data sets. Each data set is stored in a separate text file. This was done for convenience, because in this way the plant, compensator, and specifications sets can be used in multiple problems as long as data conformability is satisfied. The specific format of each of these files is described in detail in Section 3.

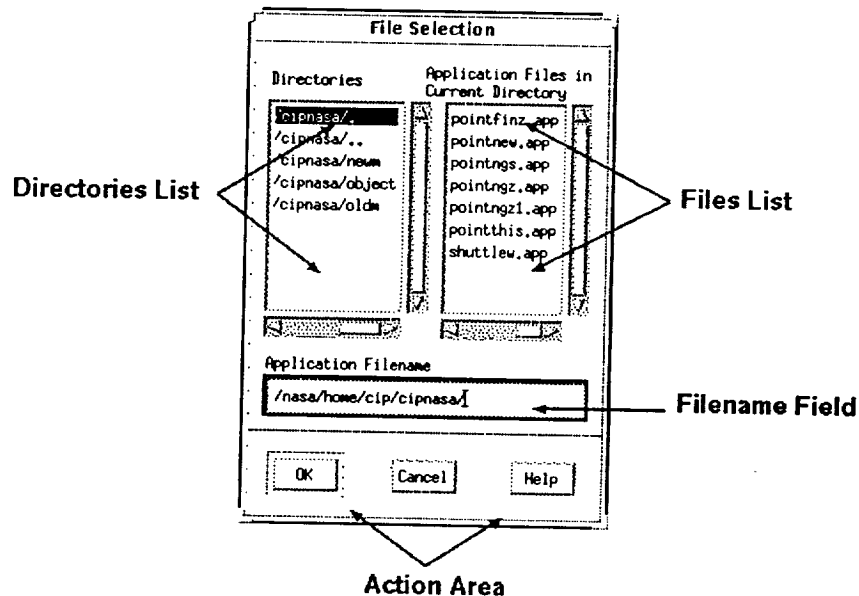


Figure 3 File Selection Box

OUCIP can input these data files in two ways. Each of the four input files can be chosen from a corresponding selection box which is obtained by going to the *File* menu, selecting the *New*, *Plant*, *Compensator*, or *Specifications* option, and choosing the *From Text File* option in the submenu that comes up. Each selection will create a file selection box similar to that shown in Figure 3, from which a file can be chosen. (Several ways of selecting options from menus are explained in Section 3.)

Data from the four files can also be entered by means of the *Application/Retrieve* feature, which is also an option in the *File* menu. This feature allows the user to enter the name of a file in which a title for the problem and the names of the four data files are given. (See Section 3 for exact format of the file). When using this feature, the user must make sure that all the files named in the *.app* file exist and contain the necessary data.

The same procedures above (file by file or application file) can be used to save data sets. The settings, plant, compensator, or specifications file can be saved individually by choosing the appropriate option in the *File* menu. By using the *Application/Save* option, *OUCIP* will create (or overwrite) and write to five files called X.app (problem title and names of the other four files), X.set (settings), X.plt (plant), X.cmp (compensator), and X.spc (specifications), where X is the base name given by the user in the *Application/Save* File selection box. For example, if the filename temp.app is given in the selection box, files temp.app, temp.set, temp.plt, temp.cmp, and temp.spc will be created. If these files already exist and the user chooses to overwrite the application file X.app, each of the five files is backed up before overwriting, following the unix convention of appending a percent sign (%) to the original filename to form the name of the backup file.

The file extensions .app, .set, .plt, .cmp, and .set are very important for *OUCIP*. For example, the file selection box for loading a plant data file will only list files that end in .plt. A file saved with an extension that does not correspond to the data in the file according to the convention given above will either be ignored by *OUCIP* directory listings or will generate errors during data loading.

NOTE: All input files must exist in the current directory (the directory from which *OUCIP* is executed).

3.0 Graphical User Interface and Data Files Formats

OUCIP is equipped with a graphical user interface (GUI) that makes it user friendly. The main menu is comprised of the items *File*, *Parameters*, *Execute*, *Graphics*, *Specifications*, *Activate*, and *Help*. Except for the *Help* button, each of these items has a pulldown menu that is mapped to the screen when the item is selected with the first mouse button or by pressing *F10* and the corresponding mnemonic (*F* for *File*, etc.) when the *OUCIP* main window has the keyboard focus¹. Once a pulldown menu is visible, it is possible to traverse through possible options by moving the pointer while keeping the first mouse button depressed. The corresponding mnemonic can also be used to select a visible option. The arrow keys can also be used to traverse through some of the menu options. The *enter* key can also select an option that has been made active (i.e., an option that appears raised or highlighted on the screen)². Online help is not available in this version of the program. A description of the available menus and submenus follows.

3.1 *FILE* Menu

The file menu controls the flow of non-graphical data to and from *OUCIP*. This menu has several options for retrieving and saving various types of data, which are described below. In all of Section 3, each filled or hollow bullet represents a menu option or a button in a dialog box.

3.1.1 *Settings*

- *From Text File* Retrieve settings from a text file
- *To Text File* Save settings to a text file

A settings file contains data that define the characteristics of the system and some parameters that control the behavior of the search algorithm. This text file must have a name with extension *.set* and must have the following format^{3,4}:

-
- ¹ A window has the keyboard focus when it receives all keyboard events. Depending on the window manager settings, a window is given keyboard focus by moving the pointer inside the window or by pressing the left mouse button while the pointer is on the window.
- ² A menu option can be highlighted by using the mouse, the keyboard, or a combination of the two. For more information on how to interact with the GUI, see [6].
- ³ In all file format descriptions given in this document, all words in italics are variables in the file. Every line with words in italics corresponds to a line in the corresponding data file. All words and lines not italicized are comments, explanations, directives, or possible values for the variables. Data values should be in floating point format unless otherwise noted.

Mode:

Text characters TIFR (total improvement frequency response) or SIFR (sum improvement frequency response).

TIFR requires all active constraints to improve from one iteration to the next. SIFR requires that the sum of improvements in constraints is greater than the sum of degradations. If left blank or if any other characters are entered, the default SIFR is used. SIFR is recommended for most cases. In most cases, the SIFR mode should produce acceptable results. TIFR is highly restrictive and should be used only in those cases where no degradation in a performance requirement can be tolerated. SIFR can produce small degradations in some performance measurements from iteration to iteration. However, over several iterations SIFR will usually counter-balance these degradations.

ID: Text characters (max 29 characters) that define a name to identify the problem. **No spaces are allowed**, but underscores can be used to separated words.

Tsamp, Nzero1, Nzero2, Npole1, Npole2, Key

Tsamp: sampling time for discrete time systems and zero for continuous time systems. This can be floating point or integer.

Nzero1: integer value that determines whether first order zeros of the compensator elements are constrained to the left half plane (or inside the unit circle for z-domain designs). A value of one means unconstrained; any other value means constrained.

Nzero2, Npole1, Npole2:

integer values that are similarly defined as Nzero1, but for second order zeros, first order poles, and second order poles, respectively.

Key: integer value that determines whether or not stability of the closed loop system is checked. The default value of one makes the program check close loop stability. No checking is done if this variable is set to zero.

Kin, Kout

Kin: integer value that defines the number of plant inputs

Kout: integer value that defines the number of plant outputs

Ndist, Ndistype

Ndist: integer value that defines the number of disturbance inputs

Ndistype: integer value that defines how disturbances affect the system, as follows.

0: frequency responses from disturbances to outputs read from plant file, 1: disturbances added to plant measured outputs, Y, 2: disturbances added to plant control inputs, X.

Stpmax, Stpmin, Pinact

Stpmax:

floating point number for the maximum step size allowed in the search algorithm, recommended to be no larger than 10% of the magnitude of the smallest nonzero initial compensator coefficient.

Stpmin:

floating point number for the minimum step size allowed in the search algorithm, recommended to be roughly 10% of Stpmax.

Pinact:

floating point number for the minimum difference between value of constraint and desired value before the constraint goes from active to inactive. This is also called hysteresis threshold, and it is recommended that it is between 0.01 and 0.05. Do NOT set to zero.

⁴ When manually creating a data file, all numerical data is free field, i.e., numerical values on a line should be separated by spaces or by commas.

3.1.2 Plant

- **From Text File** Retrieve plant from a text file.
- **To Text File** Save plant to a text file.

A plant file contains all necessary information about a plant. It contains numerical values that define the numbers of plant inputs and outputs and the type and number of disturbances, frequency data information, frequency responses from plant inputs to plant outputs, and the frequency responses from disturbances to plant outputs when appropriate. The sampling time is also included in this file for error checking purposes. The text file must have a name with extension *.plt* and must have the following format:

TSAMP: Floating point number for sampling time (zero for continuous time designs)
NOUT: Integer number of plant outputs
NIN: Integer number of plant inputs
NDIST, NDISTYPE: Integer number of disturbance inputs and type of disturbances (see section 3.1.1 for a definition of disturbance type)

The values of the variables above must be the same as those in the settings file loaded; otherwise there will be an error.

NPOINTS, TFREQ, UNITS:

- **NPOINTS:** Integer number of frequency points for which frequency response data is provided
- **TFREQ:** type of frequency data (text characters real or complex)
- **UNITS:** units of frequency data (text characters hz, rad, or wplane)

freq(1): First frequency

:

freq(NPOINTS): Last frequency

Real(g22(1,1,1)) Imag(g22(1,1,1))

Real(g22(2,1,1)) Imag(g22(2,1,1))

:

Real(g22(NOUT,1,1)) Imag(g22(NOUT,1,1))

:

Real(g22(NOUT,NIN,1)) Imag(g22(NOUT,NIN,1))

:

Real(g22(NOUT,NIN,NPOINTS)) Imag(g22(NOUT,NIN,NPOINTS))

Real(g21(1,1,1)) Imag(g21(1,1,1))

Real(g21(2,1,1)) Imag(g21(2,1,1))

:

Real(g21(NOUT,1,1)) Imag(g21(NOUT,1,1))

:

Real(g21(NOUT,NDIST,1)) Imag(g21(NOUT,NDIST,1))

:

Real(g21(NOUT,NDIST,NPOINTS)) Imag(g21(NOUT,NDIST,NPOINTS))

Here $g_{22}(i,j,k)$ is the frequency response from plant input j to plant output i for the k th frequency point, and $g_{21}(i,j,k)$ is the frequency response from disturbance input j to plant output i for the k th frequency point. Note that matrices G_{22} and G_{21} are written columnwise for each frequency point. See the *.plt* files of the demo problems for examples. Data for G_{21} does not have to be provided if the disturbances are additive signals to the plant control inputs, X , or to the measured outputs, Y .

3.1.2 Compensator

- **From Text File** Retrieve compensator from a text file
- **To Text File** Save compensator to a text file

A compensator file contains the dimensions and variables that define the elements of the compensation transfer function matrix. A z-domain compensator is used here for explanation purposes, but the compensator can be given in the w, or s domain. The ij-th element of the compensator transfer function matrix is assumed to have the following form:

$$G_{ij}(z) = (GAIN_{ij}) \frac{\prod_{l=1}^{N1} (ZA_l + ZB_l z) \prod_{l=1}^{N2} (ZC_l + ZD_l z + ZE_l z^2)}{\prod_{l=1}^{M1} (PA_l + PB_l z) \prod_{l=1}^{M2} (PC_l + PD_l z + PE_l z^2)} \quad (1)$$

A compensator file contains the dimensions and variables that define a compensator. The text file must have a name with extension *.cmp* and must have the following format:

NOUTC, NINC Integers that define the number of compensator outputs and inputs, respectively. These dimensions must be conformable with those of the plant file loaded.

TYPE, PLANE *TYPE* defines the format assumed for the element of the compensator matrix. Currently, *OUCIP* only handles compensators in the factored form shown in equation (1). Therefore *TYPE* should be the text data *FACTOR*. If other value is entered, an error will occur.

PLANE: character z,w,or s, to indicate the plane in which the compensators are presented. NO DEFAULT VALUE IS ASSUMED.

The remaining data in this file should be generated according to the following FOR loops.

```
FOR I=1 TO NOUTC
  FOR J=1 TO NINC
    GAIN(I,J), N1(I,J), N2(I,J), M1(I,J), M2(I,J), KONT(I,J)
    N1(I,J), N2(I,J): integer number of first order and second order terms in the numerator
                      of compensator element I,J, respectively
    M1(I,J), M2(I,J): same as N1 and N2, but for the denominator of compensator element I,J.
    KONT(I,J) = 1 means that the dc gain of this compensator element is allowed to vary
    KONT(I,J) = 2 means that the dc gain of this compensator element is not allowed to vary
    ZA(I,J,L), ZB(I,J,L), L=1 TO N1(I,J):
      zeroth and first order coefficient, respectively, of each of the N1(I,J) first order
      factors in the numerator of the I,J-th element of the compensator matrix
    ZC(I,J,L), ZD(I,J,L), ZE(I,J,L), L=1 TO N2(I,J):
      zeroth, first, and second order coefficient, respectively, of each of the N2(I,J)
      second order factors in the numerator of the I,J-th element of the compensator
      matrix
    PA(I,J,L), PB(I,J,L), L=1 TO M1(I,J):
      same as ZA and ZB, except that it is for denominators
    PC(I,J,L), PD(I,J,L), PE(I,J,L), L=1 TO M2(I,J):
      same as ZC, ZD, and ZE, except that it is for denominators
  END J
END I
```

3.1.4 Specifications

- **From Text File** Retrieve specifications from a text file
- **To Text File** Save specifications to a text file

Several types of design specifications can be improved by *OUCIP*:

- **Relative Stability**

These are open loop frequency response performance measurements obtained from each loop when it is broken between the compensator and the plant, and all other loops are closed.

- **Gain Margins**

A gain margin is a measure of the distance from the $-1+j0$ point to a $\pm 180^\circ$ crossing of a broken loop frequency response. The classical definition is as follows: a gain margin is defined as the inverse of the magnitude at the frequency for which the broken loop frequency response crosses the negative real axis in the complex plane. The classical definition provides the amount of pure gain change that must be made to produce instability. Of course this assumes that the closed loop system is stable.

- **Phase Margins**

A phase margin is defined as the amount of phase lag that must be added or subtracted at a frequency for which a broken loop frequency response magnitude is unity to produce instability. If the broken loop phase at the unity magnitude point lies in $[0, -180^\circ]$ the phase margin must be subtracted; otherwise it should be added.

- **Stability Margins**

A stability margin is defined as a closest approach of a broken loop frequency response to the $-1+j0$ point on a polar plot.

- **Attenuation Levels**

Peaks of a broken loop magnitude frequency response.

- **Closed Loop Disturbance Rejection**

Both peak and/or rms values of any element of the closed loop frequency response matrix from disturbance inputs to measured outputs can be specified to be below a desired level.

- **Compensator DC gains**

It is possible to specify that the DC gain of the elements of the compensator frequency response matrix should be greater than or equal to required values. Increasing compensator DC gains will improve steady-state and disturbance rejection characteristics of a closed loop system. (The term DC gain is used loosely here. In a type one or type two open loop system the appropriate term should be velocity constant and acceleration constant, respectively.)

- Compensator Damping Ratios

By specifying minimum damping ratios for the second order terms of numerators and denominators of elements of the compensator transfer function matrix it is possible to prevent excessive peaking or notching in the compensator frequency responses. This tends to make the closed loop system more robust to plant data errors.

A specifications file contains the design specifications. Depending on the type of specification, frequency ranges for searches and for defining the specifications must be defined. For example, frequency ranges for determination of gain and phase margins are necessary, but damping ratio specifications only need the compensator input and output numbers and the desired value. Frequency values are always given in rad/sec in this file. Even in the case of w-plane compensator design, the frequencies in this file **must be real frequencies, not w-plane frequencies**. The text file must have a name with extension .spc and must have the following format:

NG, NP, NS, NA

Integer values that define the number of gain margin, phase margin, stability margin, and attenuation level specifications, respectively. This allows the user to define different values of specifications for different frequency ranges. An important consideration here is that the **frequencies** of definition for each type of specification **must be in ascending order**, e.g., $GMF(1) \leq GMF(2)$, etc.

GMF(1), GMR(1), ..., GMF(NG), GMR(NG)

gain margins are desired to be at least GMR(i) for all frequencies above GMF(i)

PMF(1), PMR(1), ..., PMF(NP), PMR(NP)

phase margins are desired to be at least PMR(i) for all frequencies above PMF(i)

SMF(1), SMR(1), ..., SMF(NS), SMR(NS)

minimum distances from broken loop frequency responses to the $-1+j0$ point are desired to be at least SMF(i) for all frequencies above SMF(i)

ASF(1), ASR(1), ..., ASF(NA), ASR(NA)

maximum magnitude of broken loop frequency responses are desired to be at most ASR(i) for all frequencies above ASF(i)

F1, F2, F3, F4, F5, F6, F7, F8

These define frequency ranges over which searches of the various margins will be done, e.g., if a user wants to phase stabilize below a frequency w_{cutoff} and gain stabilize above this frequency, the search for gain margins and phase margins can be limited to the range $[0, w_{cutoff}]$.

search for gain margins between frequencies F1 and F2

search for phase margins between frequencies F3 and F4

search for stability margins between frequencies F5 and F6

search for attenuation levels between frequencies F7 and F8

ze, zemdr zemdr is the minimum damping ratio specification for compensator zeros. Here ze is not a variable, but the characters 'z' and 'e'.

po, pomdr pomdr is the minimum damping ratio specification for compensator poles. Again, po here is literally 'po' and not a variable.

All fields and lines described above must exist in a .spc file. Optional lines (any number of them and in any order) defining the compensator D.C. gain specifications and the closed loop disturbance rejection specifications can be placed in the file as long as they have one of the following forms. (*Note:* dc, p, or r are characters that must be typed in at the beginning of each line so that *OUCIP* can correctly interpreted the data that follows):

<i>dc, nout, nin, value</i>	minimum D.C. gain for the (nout,nin) element of the compensator t.f. matrix
<i>p, nout, nin, value</i>	maximum peak value for the (nout,nin) element of the frequency response matrix from disturbances to measured outputs ($G_{2i}(jw)$).
<i>r, nout, nin, value</i>	maximum RMS value for the (nout,nin) element of $G_{2i}(jw)$.

The variables nout and nin must be given integer values.

3.1.5 Application

- **Retrieve** Retrieve application status from a text file
- **Save** Save application status to a text file (see Section 2)

An application file contains the name (title) of the system and the names of a specifications file, a plant file, a compensator file, and a settings file. This option can be used in two ways. First, it is possible to save the status of an application so that the design can be interrupted and saved for retrieval and continuation at a later time. It is also possible, when starting with a new design, to create an application file with the necessary filenames so that all the design data can be input from a single selection, instead of having to go through four selection boxes to input settings, plant, compensator, and specifications. The text file must have a name with extension *.app* and must have the following format:

PROBLEM_TITLE :	A title for identifying the problem. Must not include spaces.
specifications filename :	Name of the file containing the specs. Must end with <i>.spc</i>
plant filename :	Name of the file containing the plant. Must end with <i>.plt</i>
compensator filename :	Name of the file containing a compensator. Must end with <i>.cmp</i>
settings filename :	Name of the file containing settings. Must end with <i>.set</i>
diary output filename:	A filename for optional output of diary information.
screen log filename:	A filename for optional output of screen log information. For more information on diaries and screen logs, please see the next subsection.

NOTE: All input files must exist in the current directory (the directory from which *OUCIP* is executed).

3.1.6 Text Logs

This option allows the user to save in files the history of a compensator improvement session. At this time three files can be saved. The options under the *Text Logs* menu are as follows.

- **Diary**
Compensator, broken loop frequency responses, and values of the relative stability objective functions at any iteration of the search algorithm, or any combination of them, can be sent to this file.

- **Filename**
Dialog box for input of diary filename.
- **Save to Diary (ON or OFF)**
Activates or deactivates the writing of information to diary file.
- **Compensator (ON or OFF)**
Activates or deactivates output of current compensator to diary file.
- **Frequency Responses (ON or OFF)**
Activates or deactivates output of broken loop frequency responses to diary file.
- **Objective Functions (ON or OFF)**
Activates or deactivates output of relative stability constraint values to diary file.
- **Screen Log**
This file will contain almost all text that *OUCIP* sends to the screen from which the program was started (or to the console window if the program was started from a file manager), as long as the user chooses to write to the file.
 - **Filename**
Dialog box for input of screen log filename.
 - **Save to Screen Log (ON or OFF)**
Activates or deactivates the writing of information to screen log file.
- **Current Iteration**
 - **Filename**
Dialog box for input of filename for current compensator information.
 - **Write current compensator now**
Immediately write a non-reusable file with information about the current compensator.

Some of the information in the *Diary* and *Screen Log* files is the same. Some of the information, however, will only be output to one of the two files. Text can be saved to both files at the same time. If information about dc gains, damping ratios, and/or closed loop disturbance rejection is desired, the *Screen Log* option must be activated. If compensator coefficients and/or broken loop frequency responses are needed, the *Diary* option must be activated.

- 3.1.8 Quit** Prompt the user for a decision about termination of execution of *OUCIP*. This option can also be selected by typing Ctrl-C when the *OUCIP* main window has the keyboard focus. If *OUCIP* is run in the foreground from a command window, then typing Ctrl-C when that command window has the keyboard focus will kill the program without any warnings and all execution data that has not been saved will be lost.

3.2 PARAMETERS Menu

The *Parameters* menu allows the user to change the name of the problem and several settings that control the search algorithm. Each option will map to the screen a simple dialog box comprised of a text field, an *O.K.* button and a *CANCEL* button. Because of its simplicity, this dialog box is not presented here. The variables that can be changed from this menu are:

3.2.1 Title

Name of the problem. This is useful for identifying the problem in text output.

3.2.2 Mode

TIFR (total improvement frequency response) or SIFR (sum improvement frequency response). For definition of these terms, see *File/Settings* option. SIFR mode is recommended for most applications.

3.2.3 Step Size

A number that determines the amount by which the compensator parameters are changed at an iteration of the search algorithm.

3.2.4 Maximum Step Size

The step size will never be larger than this user-specified value. This is done in order to ensure that the assumptions underlying the algorithm are reasonably satisfied. The algorithm used by *OUCIP* assumes that the changes in the performance metrics are linear with respect to the compensator coefficients. If the changes in the coefficients are too large, this assumption can be grossly violated and the results can be surprising. As a rule of thumb, the maximum step size should be initially set conservatively, e.g., 10% of the magnitude of the smallest compensator coefficient that will be varied. The user should then watch the progress of the design process; if the maximum step size is being used at every iteration, this can probably be doubled.

3.2.5 Minimum Step Size

The step size will never be smaller than this user-specified value. This prevents the iterative algorithm from "jamming" or reaching a point in which the change in the coefficients produces no practical improvement in any performance metric. In some cases, when the active constraints are conflicting, the program may tend to reduce the step until an "impasse" is reached or the minimum step setting is violated. It is possible in some instances that performance improvement can be achieved by resetting the step to a higher value. In such cases the larger step size allows the algorithm to move from a local impasse region to a region suitable for continued improvements in the performance metrics. In other cases, it may be necessary to change the types and/or number of activated specifications. This is better understood by realizing that the

performance metrics are very nonlinear functions of the compensator coefficients. When the algorithm converges, microscopic changes will usually have little effect on future improvement in the performance measurements. However macroscopic step perturbations can often make significant differences that allow the algorithm to escape from local impasse regions.

3.2.6 Hysteresis Threshold

An active constraint (unsatisfied performance metric) will be automatically made inactive when its value is better than the design specification by an amount larger than or equal to the hysteresis threshold. However, a constraint becomes active when it violates the desired specification. This eliminates the tendency of constraints to bounce from active to inactive and viceversa on consecutive iterations. This should be a small positive value, but **NOT zero**. Values between 0.01 and 0.05 have been found to be suitable.

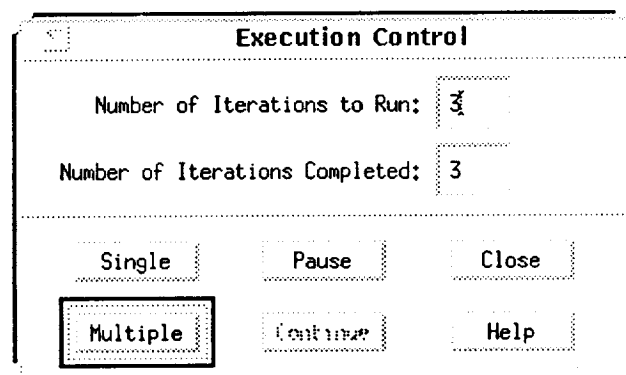
3.3 EXECUTE Menu

The search algorithm is started, stopped, paused, and continued from this menu. The available options are:

3.3.1 Single Run one iteration of the search algorithm

3.3.2 Dialog

Create a dialog box from which execution of a single or multiple iterations can be run. This dialog box is comprised of two areas, as shown in Figure 4. The first area contains a text field to enter the number of iterations to run and a text field that displays how many of the iterations entered in the first text field have been run. The second area contains six buttons whose actions are described as follows.



The dialog box is titled "Execution Control" and has a standard Windows-style title bar with a close button. It is divided into two main sections by a horizontal dotted line. The top section contains two text input fields: "Number of Iterations to Run:" with the value "3" entered, and "Number of Iterations Completed:" with the value "3" entered. The bottom section contains six buttons arranged in two rows of three. The buttons are labeled "Single", "Pause", "Close", "Multiple", "Continue", and "Help". The "Multiple" button is highlighted with a thick black border.

Figure 4 Execution Control Dialog Box

- **Single** run one iteration
- **Multiple** run the number of iterations given in the first text field of the dialog box
- **Pause** pause after current iteration when running multiple iterations
- **Continue** complete the number of iterations given in the first text field
- **Close** kill the execution dialog box
- **Help** no online help is available in this version of *OUCIP*

Depressing the return key while the execution dialog box has the input focus will have the same effect as pressing the button that is currently highlighted.

3.3.3 *Pre/Post Analysis*

When no iterations have been run, this option invokes the **Pre-analysis** feature of *OUCIP*, which computes frequency responses and values of the constraints without applying any changes to the compensator. When one or more iterations have been run, this option invokes the **Post-analysis** feature, which evaluates all constraints and frequency responses with the compensator obtained in the most recent iteration. The pre-analysis feature allows the user to compute what the initial compensation produces and can be used to help set initial design requirements. The post-analysis feature can be used to save the results from the last iteration to log files if saving of text to log files was inactive during program execution.

3.4 *GRAPHICS Menu*

The graphics menu allows the user to create windows and assign their contents, clear and destroy windows, and assign default window sizes. The available options are:

3.4.1 *Create Window*

Maps to the screen a window creation dialog box. This dialog box, as shown in Figure 5, is comprised of five areas:

- **Matrix Selection**
Toggle box for selection of the frequency response matrix from which data is to be plotted. The possible choices in this version of *OUCIP* are: plant, compensator, broken loop, closed loop frequency response matrix from u (same as r) to y , closed loop frequency response from d to y , determinant of the return difference matrix, and minimum singular value of the return difference matrix.

- **Element Selection**

Text fields for selection of a desired element in the frequency response matrix. In the case of broken loop frequency responses, this is given by a line number; in all other cases the element is given by output and input numbers. In the case of the determinant or the minimum singular value of the return difference matrix, which are scalar quantities, line or input/output numbers are not required.

- **Plot Type Selection**

Toggle box for selection of the type of plot: magnitude, phase, magnitude and phase (two windows), or polar plot.

- **Plot Labels**

Text fields for the title, horizontal, and vertical labels of the plot. The user can type in any desired labels here. A default plot title can be entered by using the *Default Title* button as explained below.

- **Action Area**

Create Window

Will create the plot window(s).

Close

Close the plot creation dialog.

Default Title

Change value of the plot title text field to an appropriate default value.

Frequency Response Plot Definition

Line Number:
 Input Number:
 Output Number:

☒ Magnitude
 ☐ Phase
 ☐ Both
 ☐ Polar

Plot(s) Title:

Frequency Axis Label:

Magnitude Axis Label:

Phase Axis Label:

Figure 5 Plot Creation Dialog Box

3.4.2 *Kill Window*

When this option is selected, the user will be asked to delete one window by clicking on it. Hitting any key at this moment cancels the window destruction procedure. Windows can also be killed by means of the window manager.

3.4.3 *Window Size*

Selection of default window size to be used at creation of all subsequent windows.

3.4.4 *Clear Windows*

Clear all windows while retaining information about desired contents.

3.4.5 *Kill All Windows*

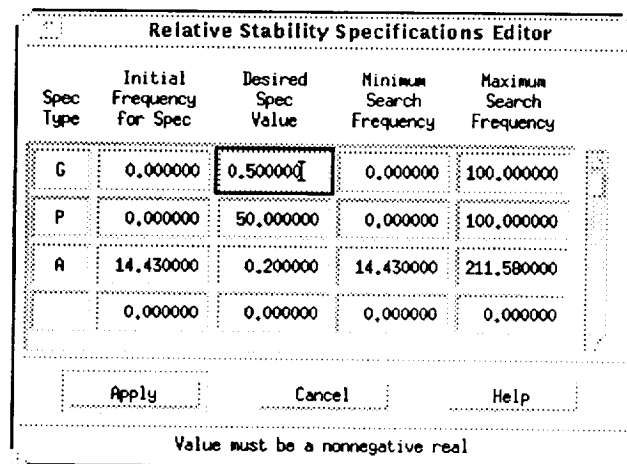
All plot windows are killed.

3.5 *SPECIFICATIONS* Menu

The specifications menu allows the user to change the specifications online. The options are:

3.5.1 *Relative Stability*

Create a dialog box in which relative stability specifications can be created or edited. An instance of this dialog box is shown in Figure 6.



The dialog box titled "Relative Stability Specifications Editor" contains a table with five columns: "Spec Type", "Initial Frequency for Spec", "Desired Spec Value", "Minimum Search Frequency", and "Maximum Search Frequency". The table has four rows of data. The first row has "G" in the first column, "0.000000" in the second, "0.500000" in the third (which is highlighted with a black border), "0.000000" in the fourth, and "100.000000" in the fifth. The second row has "P", "0.000000", "50.000000", "0.000000", and "100.000000". The third row has "A", "14.430000", "0.200000", "14.430000", and "211.580000". The fourth row has empty cells for the first three columns and "0.000000" for the last two. Below the table are three buttons: "Apply", "Cancel", and "Help". At the bottom of the dialog box, there is a text label: "Value must be a nonnegative real".

Spec Type	Initial Frequency for Spec	Desired Spec Value	Minimum Search Frequency	Maximum Search Frequency
G	0.000000	0.500000	0.000000	100.000000
P	0.000000	50.000000	0.000000	100.000000
A	14.430000	0.200000	14.430000	211.580000
	0.000000	0.000000	0.000000	0.000000

Apply Cancel Help

Value must be a nonnegative real

Figure 6 Relative Stability Specs Dialog Box

Gain, phase, and stability margins and attenuation levels, their corresponding frequencies of definition and the frequency ranges for searches of margins can be edited in this dialog box. New specifications can be added at the bottom of the existing list. After pressing the apply button, each type of specification will

be automatically sorted in order of ascending frequencies. If the user desires to verify the correct order of specifications given to *OUCIP*, this dialog box should be mapped to the screen again.

3.5.2 Disturbance Rejection

- **Peaks**

Create a dialog box in which maximum values for the peaks of the frequency responses from disturbances to measured outputs can be created or edited. Figure 7 shows an instance of this dialog box. Each row corresponds to a measured output; each column corresponds to a disturbance input.

	d1	d2	d3	d4	d5
y1	-0.007000	-1.000000	0.000000	0.000000	0.000000
y2	-1.000000	0.007000	0.000000	0.000000	0.000000
y3	0.000000	0.000000	0.000000	0.000000	0.000000
y4	0.000000	0.000000	0.000000	0.000000	0.000000

Value must be positive real or -1 for no specification

Figure 7 Peaks Specifications Dialog Box

- **RMS**

Create a dialog box in which maximum RMS values for the frequency responses from disturbances to measured outputs can be created or edited. This dialog box is similar to that shown in Figure 7.

3.5.3 Compensator DC Gain

Create a dialog box in which minimum values for the DC gain of desired compensator elements can be created or edited. This dialog box is similar to that of Figure 7, with the exception that rows and columns are labeled according to compensator output and input numbers, respectively.

3.5.4 Compensator Zeroes Damping Ratios

Create a simple dialog box to create or change desired minimum damping ratios for second order terms in the numerators of all compensator elements.

3.5.5 Compensator Poles Damping Ratios

Create a simple dialog box to create or change desired minimum damping ratio for second order terms in the denominators of all compensator elements.

3.6 *ACTIVATE* Menu

This menu allows the user to activate or deactivate the different specifications to be improved. The available options are:

3.6.1 *Relative Stability*

Activate or deactivate gain, phase and stability margins, and attenuation levels.

3.6.2 *Disturbance Rejection*

Activate or deactivate disturbance rejection (peak and rms) specifications.

3.6.3 *Compensator DC Gain*

Activate or deactivate compensator DC gain specifications.

3.6.4 *Compensator Damping Ratios*

Activate or deactivate compensator damping ratio specifications.

3.7 *HELP* Menu

Online help is not available in this version of *OUCIP*.

3.8 *OUCIP* Plots

One of the important features of *OUCIP* that was not in the original compensator improvement program is the ability to monitor the progress of the search algorithm by means of graphics. Section 3.4.1 describes the plot creation dialog, the variables that can be plotted, and the types of plots that can be obtained. This section provides some additional details about each type of plot. Hard copies can be obtained by using any screen dump utility. High resolution hard copies, e.g., in the Postscript language, are not available in this version of *OUCIP*. In all plot windows, red lines are used to plot results of the current iteration. Blue lines represent results from the previous iteration. Results from the first iteration for which the window existed are always plotted with orange lines.

3.8.1 Semilog Plots

Both magnitude and phase plots for all frequency responses can be created. Magnitudes are always plotted in decibels. Phase is always plotted in degrees. Frequencies are always in Hz, unless the design is done in the w-plane, in which case frequencies are in rad/sec.

When the semilog plot corresponds to the magnitude of the frequency response for a compensator element, a label shown in the lower left corner of the window will indicate the value of the DC gain of that compensator element, unless DC gain specifications are not active, in which case the label "inactive" will appear in that corner.

When the semilog plot corresponds to the magnitude of the frequency response from a disturbance input to a measured output, a label shown in the lower left corner of the window will indicate the RMS value of the corresponding frequency response magnitude, unless disturbance rejection specifications are not active, in which case the label "inactive" will appear in that corner.

3.8.2 Polar Plots (and data sweep)

Any of the frequency responses generated can be plotted in polar form. This is particularly useful for observing relative stability. Magnitudes are always in decibels, as the label in the lower left corner of all polar plots indicates.

As an aid in the interpretation of polar plots, it is possible to **sweep through the data** by clicking inside the outer circle of the plot with the left mouse button and then moving the left and right arrow keys. Pressing the first button will change the cursor to a cross-hair and place it at the closest data point available. The left and right arrow keys will make the cursor move up and down in frequency through the existing data. Magnitude, phase, and frequency for the currently selected data point are shown on the right lower corner of the window. Pressing the second mouse button inside the circle or pressing the first button in any other window will cancel the sweeping operation.

4.0 Example

For illustration purposes, *OUCIP* is applied here to improve the design of a compensator for the simple two-axis pointing system shown in Figure 8. This plant has torque control inputs T_{EL} and T_{AZ} , angular displacement outputs θ_{EL} and θ_{AZ} , and torque disturbance inputs D_{EL} and D_{AZ} . The dynamics of the system include lightly damped modes at 0.16 and 0.32 Hz and rigid body modes, and there is significant coupling between input/output pairs. A sampled data system with a sampling time of 0.1 seconds is used; the design is performed in the z-domain. The plant is described by 200 data points for each of the 8 open-loop frequency responses.

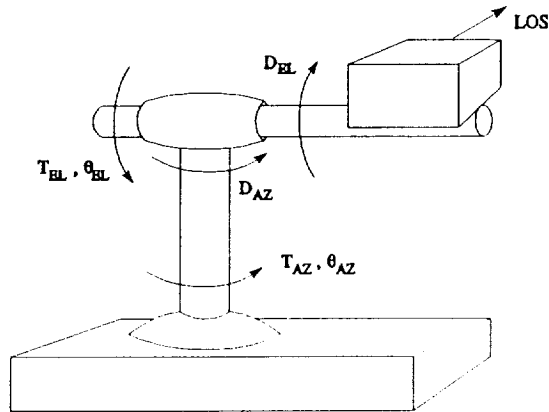


Figure 8 Two-Axis Pointing System

An initial diagonal compensator was designed by using two cascaded first order lead stages in element 1,1 and a cascade combination of a first order and a second order lead stages in element 2,2.

The application file used for this example is listed below. The names of the other files used are listed in the application file. Listings of the contents of all input files follow. In the case of the plant file, only segments of the frequency vector and frequency response data are shown because of space limitations.

```
POINTING_SYSTEM
pointngz.spc
pointngz.plt
pointngz.cmp
pointngz.set
pointngz.ou1
pointngz.ou2
```

File pointngz.app

```

SIFR
POINTING_SYSTEM
0.1,1,1,1,1,1
2,2
2,0
.1,.00000001,.002

```

File pointngz.set

```

0.1
2
2
2,0
199,real,hz
1.0000000000000000e-02
1.122595481859776e-02
1.260220615891982e-02
.
.
.
3.418616627036982e+00
3.880523059228015e+00
4.404839985304809e+00
-2.527993151923695e+01 7.938784287492003e-02
-2.030426157504917e+01 6.381326129448041e-02
.
.
.
-2.730763043920993e-06 1.428328525756445e-05
-2.800968491352494e-07 1.471377941616962e-06

```

Segments of File pointngz.plt

```

2 2
FACTOR z
7.079457843841380e-02 2 0 2 0 1
-2.288862873373321e+03 2.290862873373321e+03
-8.449471970135090e+00 1.044947197013509e+01
8.256081162284269e-01 1.174391883771573e+00
-3.632987249468415e+00 5.632987249468415e+00
0.0 0 0 0 0 2
0.0 0 0 0 0 2
3.162277660168379e-02 1 1 1 1 1
-2.288862873373321e+03 2.290862873373321e+03
5.230412981082986e+00 -1.051352552199004e+01 5.335900094654075e+00
8.256081162284269e-01 1.174391883771573e+00
1.000000000000000e+00 -1.951782415225777e+00 1.004569968972794e+00

```

File pointingz.cmp

```

1 1 0 1
0 .5
0 50
0 .1
14.43 .2
0,100,0,100,0,,1,14.43,211.58
ze,0.3
po,0.3
dc,1,1,0.511
dc,1,2,0.512
dc,2,1,0.521
dc,2,2,0.522
p,1,1,.007
r,1,1,.007
p,2,2,.007
r,2,2,.007

```

File pointngz.spc

The initial compensator does not meet all design specifications. A final compensator obtained by *OUCIP* after 220 iterations satisfied all requirements. The following segments of a screen log file saved during execution show initial and final status of the constraints. A comparison of the initial and final segments of this file shows that all performance measurements have been improved to satisfy design specifications. In this output file, the word **ACTIVATED** refers to a specification that the user has made visible to *OUCIP* by means of the **ACTIVATE** menu. The word **ACTIVE** as part of the table captions shows which of the performance measures satisfy design specifications (*NO* value in the **ACTIVE** column) and which do not (*YES* value).

The initial segment of the file shows that the first broken loop frequency response has two gain margins, both of which satisfy the specified value of 0.5, five gain crossover frequencies for which the phase satisfies the desired phase margin of 50 degrees, and two attenuation levels that satisfy the desired maximum 0.2. The second broken loop frequency response has one gain margin of 0.8851 that satisfies the specification, four gain crossover frequencies at which the phase satisfies the phase margin specification of 50 degrees, one gain crossover frequency (0.7961 Hz) at which the phase margin is 44.37 (below the desired value of 50 degrees), one attenuation level of .2170 at 2.059 Hz which does not satisfy the 0.2 maximum, and one attenuation level of .0947 at 4.405 Hz that satisfies the specification. The damping ratios of the second order numerator and denominator factors in compensator element 2,2 are both under the desired level of 0.3. The DC gains of both diagonal compensator elements are below the desired values of .511 (for element 1,1) and .512 (for element 2,2). The rms values of the closed loop frequency responses from disturbance 1 to output 1 and from disturbance 2 to output 2 are both under the specified maximum of 0.007. The closed loop frequency response from disturbance 1 to output 1 contains three peak values that violate the desired maximum of 0.007. One peak value violates the specification in the closed loop frequency response from disturbance 2 to output 2.

POINTING_SYSTEM

Thu Mar 23 14:51:57 1995

*** ACTIVATED OUCIP SPECIFICATIONS ***

RELATIVE STABILITY
 DISTURBANCE REJECTION
 COMPENSATOR DAMPING RATIOS
 COMPENSATOR DC GAINS

BROKEN LOOP 1 RELATIVE STABILITY INFORMATION

NO.	MARGIN RADIUS	FREQUENCY(Hz)	DESIRED MARGIN	MARGIN TYPE	ACTIVE
1	0.8243	0. 0.2825	0.5000	G	NO
2	0.9095	0. 3.881	0.5000	G	NO
3	140.1	0. 0.6362E-01	50.00	P	NO
4	130.1	0. 0.1303	50.00	P	NO
5	97.06	0. 0.2349	50.00	P	NO
6	75.96	0. 0.3041	50.00	P	NO
7	56.37	0. 0.7013	50.00	P	NO

ATTENUATED FREQUENCY INFORMATION

NO.	MARGIN RADIUS	FREQUENCY(Hz)	DESIRED MARGIN	MARGIN TYPE	ACTIVE
8	0.1668	0. 2.059	0.2000	A	NO
9	0.7974E-01	0. 4.405	0.2000	A	NO

BROKEN LOOP 2 RELATIVE STABILITY INFORMATION

NO.	MARGIN RADIUS	FREQUENCY(Hz)	DESIRED MARGIN	MARGIN TYPE	ACTIVE
1	0.8851	0. 3.881	0.5000	G	NO
2	125.1	0. 0.4497E-01	50.00	P	NO
3	168.8	0. 0.1238	50.00	P	NO
4	164.2	0. 0.1444	50.00	P	NO
5	114.8	0. 0.2182	50.00	P	NO
6	44.37	0. 0.7961	50.00	P	YES

ATTENUATED FREQUENCY INFORMATION

NO.	MARGIN RADIUS	FREQUENCY(Hz)	DESIRED MARGIN	MARGIN TYPE	ACTIVE
7	0.2170	0. 2.059	0.2000	A	YES
8	0.9469E-01	0. 4.405	0.2000	A	NO

COMPENSATOR DAMPING RATIO INFORMATION

OUTPUT	INPUT	FREQUENCY(Hz)	DAMPING RATIO	DESIRED MARGIN	LOCATION
2	2	0.1592	0.9984E-01	0.3000	ZERO
2	2	0.3661	0.9912E-02	0.3000	POLE

COMPENSATOR DC GAIN INFORMATION

OUTPUT	INPUT	DC GAIN	DESIRED MARGIN
1	1	0.7079E-01	0.5110
2	2	0.3162E-01	0.5220

CLOSED LOOP DISTURBANCE REJECTION INFORMATION

OUTPUT	DISTURB.	PREVIOUS RMS	RMS VALUE	DESIRED MARGIN
1	1	0.4473E-02	0.4616E-02	0.7000E-02
2	2	0.3420E-02	0.3486E-02	0.7000E-02

CLOSED LOOP DISTURBANCE REJECTION INFORMATION

OUTPUT	DISTURB.	FREQUENCY(Hz)	PREVIOUS PEAK	PEAK VALUE	DESIRED MARGIN
1	1	0.1000E-01	0.1848E-01	0.1880E-01	0.7000E-02
1	1	0.2885	0.8854E-02	0.9583E-02	0.7000E-02
1	1	0.3107	0.9387E-02	0.9584E-02	0.7000E-02
2	2	0.1000E-01	0.2198E-01	0.2252E-01	0.7000E-02

NUMBER OF ACTIVE MARGINS = 10

ITERATION 0 SUCCESSFULLY COMPLETED. Step: 0.1000000

Initial Segment of File pointingz.ou2

The final segment of the file shows that all broken loop specifications (gain margins, phase margins, and attenuation levels) have been satisfied (note the NO entries in the ACTIVE column for all these variables). In the final segment of the file all specifications are activated, but information on DC gains and damping ratios of compensator elements and closed loop disturbances to measured outputs peak values are not shown because all these specifications are satisfied. If values of these variables are needed, it is possible to change the specifications to more ambitious values so that they are violated and show in the output. Rms values of the frequency responses from disturbance 1 to output 1 and from disturbance 2 to output 2 satisfy the specification of 0.007 in the final design.

*** ACTIVATED OUCIP SPECIFICATIONS ***					

RELATIVE STABILITY					
DISTURBANCE REJECTION					
COMPENSATOR DAMPING RATIOS					
COMPENSATOR DC GAINS					

BROKEN LOOP 1		RELATIVE STABILITY INFORMATION			
NO.	MARGIN RADIUS	FREQUENCY(Hz)	DESIRED MARGIN	MARGIN TYPE	ACTIVE
1	3.543	0. 0.3460	0.5000	G	NO
2	73.62	0. 0.5586E-01	50.00	P	NO
3	157.5	0. 0.6740E-01	50.00	P	NO
4	50.00	0. 0.6178	50.00	P	NO
ATTENUATED FREQUENCY INFORMATION					
NO.	MARGIN RADIUS	FREQUENCY(Hz)	DESIRED MARGIN	MARGIN TYPE	ACTIVE
5	0.1984	0. 2.059	0.2000	A	NO
6	0.1002	0. 4.405	0.2000	A	NO
BROKEN LOOP 2		RELATIVE STABILITY INFORMATION			
NO.	MARGIN RADIUS	FREQUENCY(Hz)	DESIRED MARGIN	MARGIN TYPE	ACTIVE
1	0.8839	0. 3.881	0.5000	G	NO
2	80.74	0. 0.6092E-01	50.00	P	NO
3	131.0	0. 0.6740E-01	50.00	P	NO
4	58.50	0. 0.2026	50.00	P	NO
5	159.0	0. 0.2877	50.00	P	NO
6	71.92	0. 0.4500	50.00	P	NO
ATTENUATED FREQUENCY INFORMATION					
NO.	MARGIN RADIUS	FREQUENCY(Hz)	DESIRED MARGIN	MARGIN TYPE	ACTIVE
7	0.1958	0. 2.059	0.2000	A	NO
8	0.9709E-01	0. 4.405	0.2000	A	NO
CLOSED LOOP DISTURBANCE REJECTION INFORMATION					
OUTPUT DISTURB.	PREVIOUS RMS	RMS VALUE	DESIRED MARGIN		
1	1	0.3737E-02	0.3737E-02	0.7000E-02	
2	2	0.2073E-02	0.2072E-02	0.7000E-02	
NUMBER OF ACTIVE MARGINS = 0					
ITERATION 220 SUCCESSFULLY COMPLETED. Step: 0.1773025E-02					

Final Segment of File pointngz.ou2

Figure 9 shows a screen dump of a polar plot of the frequency response of loop 1, both with the initial compensator and with the final compensator. Figure 10 shows that the desired rejection of disturbance input 1 has been achieved at the first measured output. Figures 11 and 12 show the magnitude frequency responses of both compensator elements, for the initial and the final design. In all the plots, dashed lines represent the system with the original compensator and solid lines represent the results of applying the compensator of the last iteration of *OUCIP* to the system.

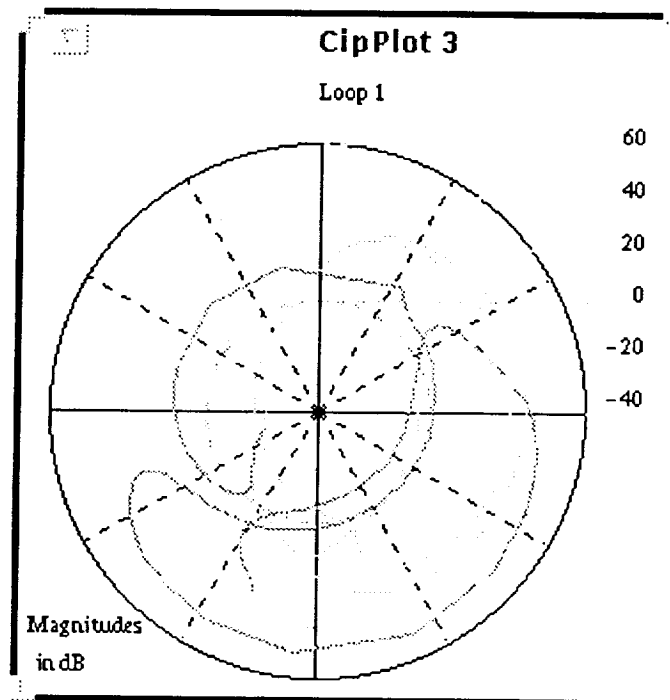


Figure 9 Frequency Response of Broken Loop 1

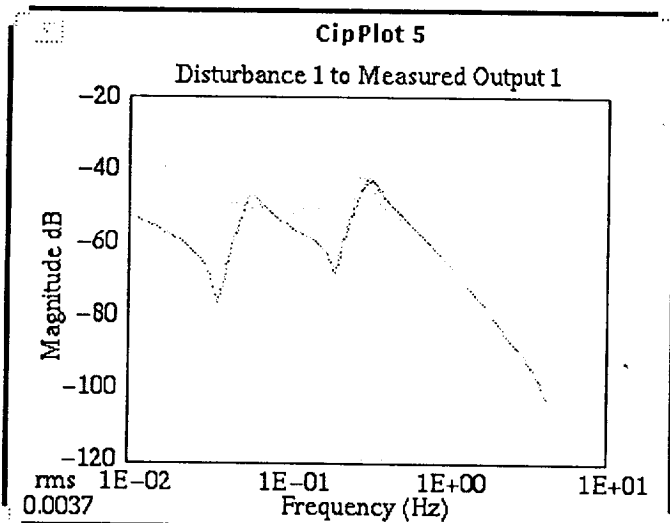


Figure 10 Magnitude: Input D_1 to Output Y_1 .

It can be observed in Figure 9 that the first broken loop has become unstable. By examining a polar plot of the determinant of the return difference matrix minus one, it can be shown that the closed loop system remains stable with the final compensator. Figure 10 shows that the peaks of the frequency response magnitudes from disturbance 1 to output 1 and from disturbance 2 to output 2 have been decreased. Figures 11 and 12 show that the low frequency gains and the damping ratios of both elements of the compensator matrix have been increased.

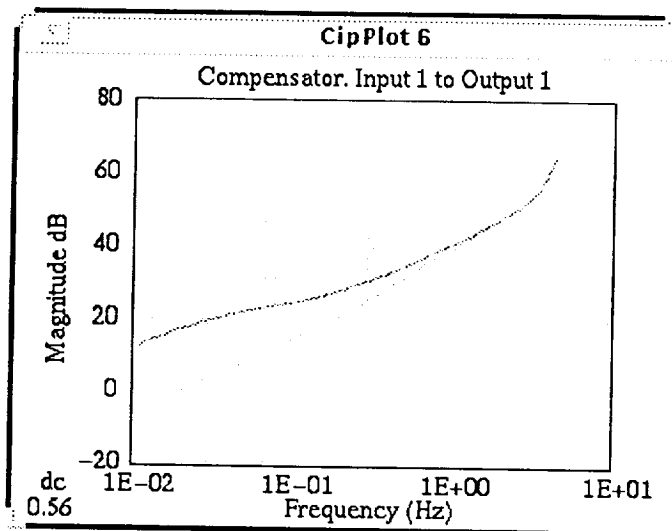


Figure 11 Compensator Element 1,1

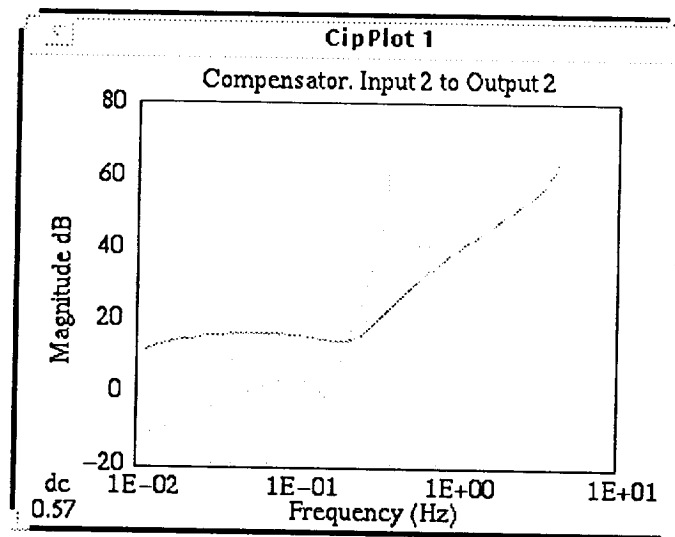


Figure 12 Compensator Element 2,2

5.0 Conclusions

Additional information regarding the Ohio University Compensator Improvement Program can be found in references [2] and [5]. Reference [2] describes the theory and associated numerical techniques for the original multi-input, multi-output CIP. Reference [5] contains the theoretical and algorithmic development behind *OUCIP*. More detailed explanations of the results of the application of *OUCIP* to several other examples are also included in [5].

6.0 References

- [1] J. R. Mitchell, W. L. McDaniel, Jr. (1973), "An Innovative Approach to Compensator Design," NASA CR-2248, May 1973.
- [2] L. L. Gresham, J. R. Mitchell, W. L. McDaniel Jr. (1980), "Multivariable Control System Design Algorithm", *Journal of Guidance and Control*, Vol. 3, No. 4, July-August 1980, pp.319-325.
- [3] M. A. Duncan (1994), "Enhancements to Ohio University's Compensator Improvement Program (OUCIP)", M.S. Thesis, Ohio University, Athens, OH, March., 1994.
- [4] E. A. Medina, M. A. Duncan, J. R. Mitchell, and R. D. Irwin (1994), "Compensator Improvement Program: New Developments and Results", *Proceedings of the Twenty-Sixth Southeastern Symposium on System Theory*, IEEE Computer Society Press, Los Alamitos, CA, March 1994, pp. 64-68.
- [5] J. R. Mitchell, R. D. Irwin, E. A. Medina, D. A. Allwine, W. G. Frazier, and M. A. Duncan (1995), *Computerized Design of Controllers Using Data Models*, Final Report, NASA Grant NAG8-217, Department of Electrical and Computer Engineering, Ohio University, Athens, Ohio 45701, July 1995.
- [6] Open Software Foundation (1990), *OSF/MotifTM User's Guide*, Revision 1.0, Prentice Hall, Englewood Cliffs, New Jersey.
- [7] D. Heller (1991), *Motif Programming Manual*, O'Reilly & Associates, Inc., Sebastopol, CA, 1991.
- [8] Andrew S. Glassner, Editor (1990), *Graphics Gems*, Academic Press, Cambridge, MA, 1990.

Appendix

The graphical user interface (GUI) for *OUCIP* was written in C working with the X Window System, release X11R5, using the Motif Toolkit, release 1.2.4. Some of the GUI modules are based on examples included in [7]. The following is the Copyright notice that accompanies all programs in reference [7].

```
/* Written by Dan Heller. Copyright 1991, O'Reilly & Associates.
 * This program is freely distributable without licensing fees and
 * is provided without guarantee or warranty expressed or implied.
 * This program is -not- in the public domain. */
```

Loose labels for plots were implemented by using a C routine written by Paul Heckbert, which can be found in [8]. The following note accompanies the software of reference [8].

The authors and the publisher hold no copyright restrictions on any of these files; this source code is public domain, and is freely available to the entire computer graphics community for study, use, and modification. We do request that the comment at the top of each file, identifying the original author and its original publication in the book *Graphics Gems*, be retained in all programs that use these files.

The matrix widget *XbaeMatrix*, version 3.8, was used in some dialog boxes. The Copyright status of the *XbaeMatrix* package is given by the following note. The release of *XbaeMatrix* used included third party modifications, as written at the end of the note.

```
/*
 * Copyright(c) 1992 Bell Communications Research, Inc. (Bellcore)
 * All rights reserved
 * Permission to use, copy, modify and distribute this material for any purpose and without fee is
 * hereby granted, provided that the above copyright notice and this permission notice appear in
 * all copies, and that the name of Bellcore not be used in advertising or publicity pertaining to
 * this material without the specific, prior written permission of an authorized representative of
 * Bellcore.
 *
 * BELLCORE MAKES NO REPRESENTATIONS AND EXTENDS NO WARRANTIES, EXPRESS OR
 * IMPLIED, WITH RESPECT TO THE SOFTWARE, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR ANY PARTICULAR
 * PURPOSE, AND THE WARRANTY AGAINST INFRINGEMENT OF PATENTS OR OTHER
 * INTELLECTUAL PROPERTY RIGHTS. THE SOFTWARE IS PROVIDED "AS IS", AND IN NO
 * EVENT SHALL BELLCORE OR ANY OF ITS AFFILIATES BE LIABLE FOR ANY DAMAGES,
 * INCLUDING ANY LOST PROFITS OR OTHER INCIDENTAL OR CONSEQUENTIAL DAMAGES
 * RELATING TO THE SOFTWARE.
 *
 * MatrixWidget Author: Andrew Wason, Bellcore, aw@bae.bellcore.com
 *
 * Modification History: David Boerschlein, dpb@air16.larc.nasa.gov
 * Lockheed Engineering & Sciences Company,
 * Under contract to NASA LaRC:
```

A C routine from the Xvertex software package was used for the drawing of rotated labels. The following notice describes the copyright restrictions of Xvertex.

```
/* *****  
/* xvertex 5.0, Copyright (c) 1993 Alan Richardson (mppa3@uk.ac.sussex.syma)  
*  
* Permission to use, copy, modify, and distribute this software and its  
* documentation for any purpose and without fee is hereby granted, provided  
* that the above copyright notice appear in all copies and that both the  
* copyright notice and this permission notice appear in supporting  
* documentation. All work developed as a consequence of the use of  
* this program should duly acknowledge such use. No representations are  
* made about the suitability of this software for any purpose. It is  
* provided "as is" without express or implied warranty.  
*/  
/* *****
```

Appendix F: MADCADS User's Guide

MADCADS

***Model and Data Oriented
Computer Aided Design System***

User's Guide

Daniel A. Allwine

Enrique A. Medina

Department of Electrical and Computer Engineering
Ohio University
Athens, Ohio 45701

July 8, 1995
Department of Electrical and Computer Engineering
Ohio University
Stocker Engineering Center
Athens, Ohio 45701

This work was supported in part by a grant from NASA Marshall Space Flight Center

SunOS and OpenWindows are trademarks of Sun Microsystems, Inc.
UNIX is a registered trademark of UNIX System Laboratories, Inc.
X Window System is a product of the Massachusetts Institute of Technology.
Motif is a registered trademark of the Open Software Foundation.
Postscript is a registered trademark of Adobe Systems, Inc.

Table of Contents

1.0 Introduction	F4
1.1 System Description, Design Goals and Design Process	F4
1.2 Getting Started	F6
1.3 Organization of this Guide	F8
2.0 Data Input	F8
3.0 Graphical User Interface and Data File Formats	F8
3.1 <i>FILE</i> Menu	F9
3.1.1 <i>Setup</i> F9; 3.1.2 <i>Quit</i> F10	
3.2 <i>PARAMETERS</i> Menu	F10
3.2.1 <i>Step Length</i> F10; 3.2.2 <i>Step Length Increase Factor</i> F11; 3.2.4	
<i>Epsilon Boundary</i> F11	
3.3 <i>EXECUTE</i> Menu	F11
3.3.1 <i>Single</i> F11; 3.3.2 <i>Dialog</i> F11	
3.4 <i>GRAPHICS</i> Menu	F12
3.4.1 <i>Create Window</i> F12; 3.4.2 <i>Destroy Window</i> F14; 3.4.3 <i>Window</i>	
<i>Size</i> F14; 3.4.4 <i>Clear Windows</i> F14; 3.4.5 <i>Destroy All Windows</i> F14	
3.5 <i>HELP</i> Menu	F14
3.6 <i>MADCADS</i> Plots	F14
3.7 File Formats and Organization	F15
3.7.1 <i>The setup (.set) file</i> F16; 3.7.2 <i>The Controller</i> F16; 3.7.3	
<i>Frequency Response Matrix Blocks</i> F17; 3.7.4 <i>Iteration Parameters</i> F17;	
3.7.5 <i>Frequency Data</i> F17; 3.7.6 <i>Singular Value Constraint Data</i> F18;	
3.7.7 <i>I/O Pair Magnitude Constraint Data</i> F19; 3.7.8 <i>Controller Pole</i>	
<i>Damping Ratio Data</i> F20; 3.7.9 <i>Controller Zero Damping Ratio Data</i> F20;	
3.7.10 <i>Controller Pole Damping Factor Data</i> F20	
4.0 References and Additional Information	F21
Notes	F22

MADCADS

Model and Data Oriented Computer Aided Design System User's Guide

1.0 Introduction

MADCADS is a graphically-oriented, interactive piece of software that assists in the process of compensator design for multivariable dynamic systems by applying a search algorithm to vary the parameters of an initial stabilizing compensator in order to improve the values of performance measurements. A more detailed explanation of the underlying theory of *MADCADS* is given by Mitchell, et. al. [1]. At the time of this writing, *MADCADS* has been compiled and tested under SunOS4.1.3, DELL UNIX System V Release 4.0 and Novell Unixware 1.1.2 (System V Release 4.2). The graphical user interface (GUI) was written for the X Window System, Version 11, Release 5, using the Motif™ Toolkit, version 1.2.4.

1.1 System Description, Design Goals and Design Process

The block diagram for a multi-input, multi-output, linear, time-invariant, discrete-time feedback control system is shown in Fig. 1, where $R \in \mathbb{R}^r$ is a vector of reference inputs, $U \in \mathbb{R}^q$ is a vector of forward path controller outputs, $V \in \mathbb{R}^m$ is a vector of plant inputs, $W \in \mathbb{R}^k$ is a vector of disturbance inputs, $Y \in \mathbb{R}^p$ is a vector of measured outputs used for feedback, $Z \in \mathbb{R}^l$ is a vector of other physical system outputs which are not or cannot be used for feedback, and $N \in \mathbb{R}^n$ is a vector of sensor noise inputs.

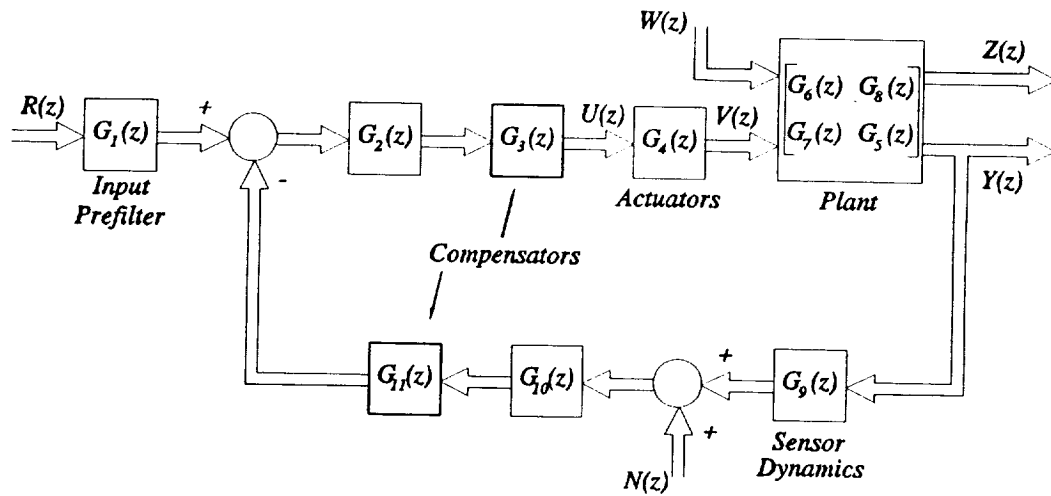


Figure 1 *MADCADS* Block Diagram

The system plant is composed of a block partitioned transfer function matrix such that $Z(z) = G_6(z)W(z) + G_8(z)V(z)$ and $Y(z) = G_7(z)W(z) + G_5(z)V(z)$. Hence, $G_6(z)$, $G_8(z)$, $G_7(z)$, and $G_5(z)$ represent the transfer function matrices from $W(z)$ to $Z(z)$, $V(z)$ to $Z(z)$, $W(z)$ to $Y(z)$, and $V(z)$ to $Y(z)$, respectively.

$G_1(z)$ represents the system reference input prefilter. $G_2(z)$ describes the dimensional scaling gains required to match the sensor and input signal dimensions to those suitable for the forward path controller (or for the actuators if no forward path controller is specified). $G_4(z)$ represents the control system actuator dynamics while $G_9(z)$ describes the sensor dynamics. $G_{10}(z)$ represents additional sensor dynamics (affected by noise) and/or dimensional scaling gains required to match the input signal dimensions of the feedback compensator.

Blocks $G_3(z)$ and $G_{11}(z)$ represent the system controllers (only one of which may be specified at a time) and must be specified in a state-variable form while all other blocks must be specified as frequency response data. The required data formats are discussed in detail in section 3.7. Also, all design is performed in the z-domain.

IMPORTANT: Blocks $G_3(z)$ and $G_{11}(z)$ cannot be included simultaneously. Only one controller may be designed at a time.

MADCADS is designed to vary the coefficients of the elements of $G_3(z)$ (or $G_{11}(z)$) so that simultaneous improvements in one or more of the following types and/or combinations of design specifications are achieved: (1) singular value frequency response, (2) I/O pair magnitude frequency response, (3) controller pole damping ratio constraints (4) controller zero damping ratio constraints (5) controller pole damping factor constraints.

IMPORTANT: Controller damping ratio and damping factor constraint handling capability has been included in this version of the software but has not been tested.

The iterative process in *MADCADS* is controlled by the user. Each iteration can be outlined as follows:

- 1) Compute the gradients of the active constraints.
- 2) Compute the search direction.
- 3) Compute the trial controller, its eigenvalues, pole damping factors, and check for controller stability. If unstable, shorten the step length and return to step 2.
- 4) Compute the trial controller pole damping ratios, frequency response, and zero damping ratios.
- 5) Calculate the trial frequency dependent constraint functions (singular value and I/O

- pair magnitude).
- 6) Check closed loop stability. If unstable, reduce the step length and go to step 2.
 - 7) Compute the new active constraints.
 - 8) Check for overall constraint improvement. If improvement is registered, increase the step length and the constraint epsilon boundary, return control to the user (or go back to step 1 if multiple iterations were requested). If no improvement was observed, decrease the step length and go to step 2.

1.2 Getting Started

MADCADS is executed by typing *madcad*s at the command prompt. The main window shown in Figure 2 will appear on the display. There are three main elements to this window: the menu bar, the message area, and the scroll bars. The menu bar allows the user to select actions to be carried out. These actions will be explained later in this document. *MADCADS* will send important execution and error messages to the message area. Information that has scrolled out of the visible portion of the message area can be seen by using the scroll bars. *MADCADS* will output information on the results of each iteration of the algorithm to the window from which it was executed or to a console window if the program is started by other means¹.

It is assumed in this guide that the user has basic knowledge about how to interact with the particular window manager being run and with software applications that use the Motif *look and feel*. Information on how to use the window manager should be available in the form of manuals that commonly accompany a workstation. A good source of information about how to use the Motif window manager and applications that run with Motif is the Motif User's Guide [2].

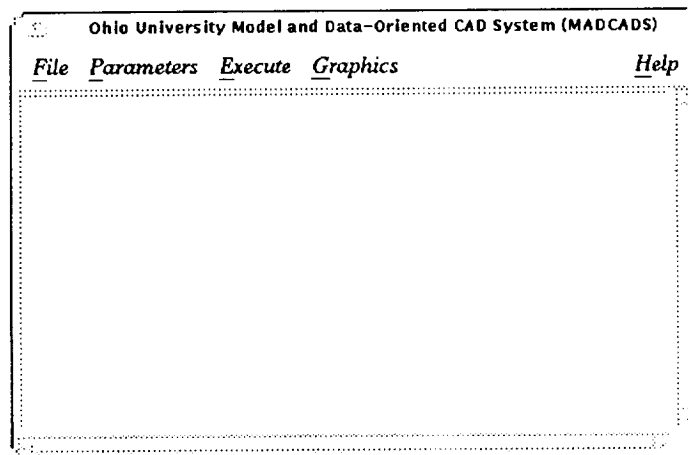


Figure 2 *MADCADS* Main Window

¹ In Openwindows™, for example, a program can be started by double-clicking on its icon in the file manager.

In order to use *MADCADS* to improve a controller design, the user must first input the necessary data to define the problem. This is done by means of the *File* menu. Then the *Execute* menu is used to start and control the iterative process. The rest of this guide explains how to create data files, modify specifications, execute iterations, and obtain text and graphical results. For the benefit of those users who want to gain some experience in using *MADCADS* before embarking on the tasks of solving problems of their own, a simple demonstration example is included with the program. The following instructions briefly describe how to load a demo problem into *MADCADS* and start the iterative process. Once a user has created the appropriate files corresponding to a problem of interest, the process is the same.

- (1) Click¹ on the *File* item of the menu bar. This will create a "pulldown" menu with two options.
- (2) Click on the *Setup* option of the pulldown menu. This will create a "cascade" menu with two options to the side of the pulldown menu.
- (3) Click on the *Retrieve...* option of the cascade menu. This will create a file selection box as shown in Figure 3.
- (4) Double-click² on the line that reads *demo.set* in the file list located on the right hand side of the file selection box. *MADCADS* will then load all the data for the demonstration example. The file selection dialog should disappear.
- (5) Click on the *Graphics* item of the menu bar and then on the *Create Window ...* option of the pulldown menu that is created. This will create a *Frequency Response Plot Definition* dialog box as shown in Figure 5.
- (6) In the top portion of the *Frequency Response Plot Definition* box, click on the toggle button next to **Rdy**. Then click on the *Create* button of the same dialog box. This will create a window with a semi-logarithmic plot of the system output return difference singular value frequency response.
- (7) Click on the *Execute* item of the menu bar. This will create a pulldown menu with two options.
- (8) Click on the *Single* option of the pulldown menu. This will run the first iteration of the design algorithm and output results to the window from which the program was started.

¹ "Click" means to press and release a mouse button without moving the pointer. Unless otherwise noted, this refers to the left mouse button in this guide.

² "Double-click" means to click a button twice in rapid succession. Unless otherwise noted, this refers to the left mouse button in this guide.

- (9) At this point, new plots can be created, and/or additional iterations can be run. The *Quit* option under the *File* menu will terminate the execution of *MADCADS*.

1.3 Organization of this Guide

The remainder of this document is divided into three sections. Section 2 explains the data needed in order to use *MADCADS* to improve a controller design. Section 3 is a reference of the graphical user interface, its various options, and the format used in the input files. Finally, Section 4 lists references and some additional information regarding *MADCADS*.

2.0 Data Input

Before the search algorithm in *MADCADS* can be used to improve a control system design, the user must create and input several sets of data. The number of data sets is directly related to the number of blocks (from Figure 1) that are needed to define the system as well as the number of files necessary to define the iteration parameters and design constraints.

First of all, a separate file is needed for each of the system blocks. As mentioned before, block $G_3(z)$ ($G_{11}(z)$) must be specified in a state variable format and the remaining blocks must be specified as matrix frequency response data. The user must also create constraint files, one each for the singular value constraints, the I/O pair magnitude constraints, the controller pole damping ratio constraints, the controller zero damping ratio constraints, and the controller pole damping factor constraints. Two additional files are also required, the first is a file that specifies the initial iteration parameters (explained in section 3.2) and the second is a file that contains a list of the above filenames for organizational purposes. This second file is the means by which *MADCADS* retrieves and saves a particular problem. The specific format of each of these files is described in detail in Section 3.

IMPORTANT: All input files must exist in the current directory (the directory from which *MADCADS* is executed).

3.0 Graphical User Interface and Data File Formats

MADCADS is equipped with a graphical user interface (GUI) that makes it user friendly. The main menu is comprised of the items *File*, *Parameters*, *Execute*, *Graphics*, and *Help*. Except for the *Help* button, each of these items has a pulldown menu that is mapped to the screen when the item is selected with the first mouse button

or by pressing *F10* and the corresponding mnemonic (*F* for *File*, etc.) when the *MADCADS* main window has the keyboard focus¹. Once a pulldown menu is visible, it is possible to traverse through possible options by moving the pointer while keeping the first mouse button depressed. The corresponding mnemonic can also be used to select a visible option. The arrow keys can also be used to traverse through some of the menu options. The *enter* key can also select an option that has been made active (i.e., an option that appears raised or highlighted on the screen)². Online help is not available in this version of the program. In this section, each bullet represents a menu option or a button in a dialog box. A description of the available menus and submenus follows.

3.1 *FILE* Menu

The file menu controls the flow of non-graphical data to and from *MADCADS*. This menu has several options for retrieving and saving various types of data, which are described below.

3.1.1 *Setup*

- *Retrieve* Retrieve problem setup from disk
- *Save* Save problem setup to disk

A setup file (with extension *.set*) contains a list of the names of the files required to define a particular design problem. A problem is loaded into *MADCADS* by first creating the necessary setup file and corresponding data sets (system block definitions, constraint definitions and iteration parameters). This activity is done outside of *MADCADS*. The user continues by starting *MADCADS* and loading the setup through the *File->Setup->Retrieve* menu selection. This will produce a file selection box where the user can choose the setup to be loaded (Figure 3). A problem can be saved for later use by using the *File->Setup->Save* menu item and then by specifying a file with extension *.set* in the resulting file selection dialog. *MADCADS* will then create the necessary data files and store them to disk. Section 3.7 contains specific information about the file formats and the manner by which file organization is accomplished through the use of the setup file.

¹ A window has the keyboard focus when it receives all keyboard events. Depending on the window manager settings, a window is given keyboard focus by moving the pointer inside the window or by pressing the left mouse button while the pointer is on the window.

² An menu option can be highlighted by using the mouse, the keyboard, or a combination of the two. For more information on how to interact with the GUI, see [6].

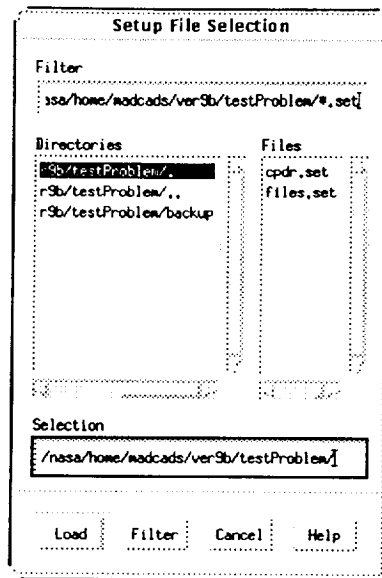


Figure 3 File Selection Box

IMPORTANT: All input files must exist in the current directory (the directory from which *MADCADS* is executed).

3.1.2 Quit Prompt the user for a decision about termination of execution of *MADCADS*.

WARNING: This option can also be selected by typing Ctrl-C when the *MADCADS* main window has the keyboard focus. If *MADCADS* is run in the foreground from a command window, then typing Ctrl-C when that command window has the keyboard focus will kill the program without any warnings and all execution data that has not been saved will be lost.

3.2 PARAMETERS Menu

The *Parameters* menu allows the user to change several settings that control the search algorithm. Each option will map to the screen a simple dialog box comprised of a text field, an *O.K.* button and a *CANCEL* button. Because of its simplicity, this dialog box is not presented here. The variables that can be changed from this menu are:

3.2.1 Step Length

A number that determines the amount by which the compensator parameters are changed at each iteration of the search algorithm. This value adjusts itself during the design process. The step length adjustment can be controlled via the *Step*

Length Increase Factor and Step Length Reduction Factor.

3.2.2 Step Length Increase Factor

A multiplicative factor by which the *Step Length* is increased after a successful iteration of the design algorithm.

3.2.3 Step Length Reduction Factor

A multiplicative factor by which the *Step Length* is decreased if an iteration fails.

3.2.4 Epsilon Boundary

Initial boundary used to determine whether or not a particular constraint is active. This value changes dynamically during the design process to prevent the number of active constraints from exceeding a pre-defined limit.

3.2.5 Armijo Factor (β)

Factor used in determination of constraint improvement. In order to register an improvement, the **actual** decrease in constraint violation (δ_x) must be greater than or equal to the product of β and the **projected** decrease in constraint violation (δ).

ie:
$$\delta_x \geq \delta \beta$$

This criterion is used to ensure that the algorithm continues to make significant progress toward its goal.

3.3 EXECUTE Menu

The search algorithm is started, stopped, paused, and continued from this menu. The available options are:

3.3.1 Single Run one iteration of the search algorithm

3.3.2 Dialog

Create a dialog box from which execution of a single or multiple iterations can be run. This dialog box is comprised of two areas, as shown in Figure 4. The first area contains a text field to enter the number of iterations to run and a text field that displays how many of the iterations entered in the first text field have been run. The second area contains six buttons whose actions are described as follows.

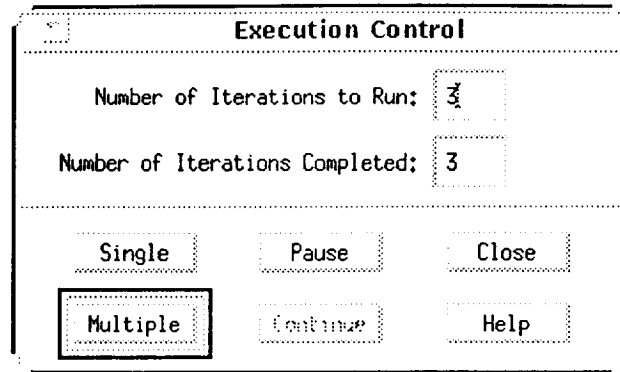


Figure 4 Execution Control Dialog Box

- **Single** run one iteration
- **Multiple** run the number of iterations given in the first text field of the dialog box
- **Pause** pause after current iteration when running multiple iterations
- **Continue** complete the number of iterations given in the first text field
- **Close** kill the execution dialog box
- **Help** no online help is available in this version of *MADCADS*

Depressing the return key while the execution dialog box has the input focus will have the same effect as pressing the button that is currently highlighted.

3.4 GRAPHICS Menu

The graphics menu allows the user to create windows and assign their contents, clear and destroy windows, and assign default window sizes. The available options are:

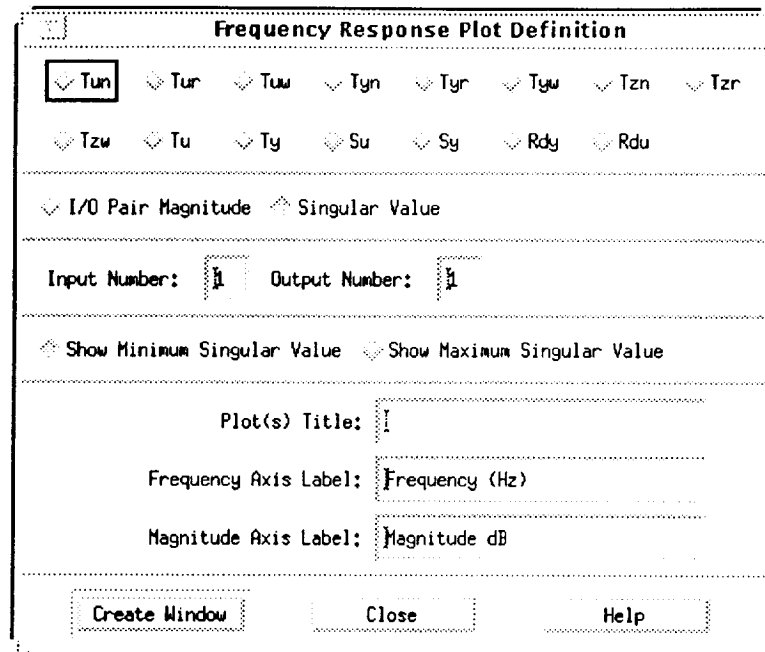
3.4.1 Create Window

Maps to the screen a window creation dialog box. This dialog box, as shown in Figure 5, is comprised of five areas:

- **Matrix Selection**
Toggle box for selection of the frequency response matrix from which data is to be plotted. The names in this area reflect the type of response and the related I/O signals from the block diagram in Figure 1. Those beginning with T are complementary sensitivity functions. For example, T_{uw} is the complementary sensitivity response from $W(z)$ to $U(z)$. Those beginning with S are sensitivity responses (ie: S_y is the system output sensitivity response). R_{du} and R_{dy} are the system output and controller output return difference matrices, respectively.

- ***Plot Type Selection***
Toggle area for selection of the type of plot: singular value or I/O pair magnitude frequency response.
- ***Element Selection***
Text fields for selection of a desired element (I/O pair) of the chosen frequency response matrix. This is only used when the *plot type selection* is "I/O Pair Magnitude."
- ***Show Minimum/Maximum Singular Value***
Toggle area for selection minimum or maximum singular value. This allows the user to look at the minimum or maximum singular value response of the selected matrix. This selection is only used when the *plot type selection* is "Singular Value."
- ***Plot Labels***
Text fields for the title, horizontal, and vertical labels of the plot. The user can type in any desired labels here. *MADCADS* will include a default label for a plot when the matrix selection is made. If any other options/selections are made **after** the matrix selection is made then the labels may no longer be appropriate for the plot. To remedy this, re-select the desired response matrix before creating the plot window (this will cause *MADCADS* to re-label the plot with the correct information).
- ***Action Area***

<i>Create Window</i>	Will create the plot window(s).
<i>Close</i>	Close the plot creation dialog.
<i>Help</i>	Not implemented.



The dialog box is titled "Frequency Response Plot Definition". It contains several sections:

- Variable Selection:** A grid of variables with checkboxes. ☒ Tun is selected. Other variables include Tur, Twu, Tyn, Tyr, Tyw, Tzn, Tzr, Tzw, Tu, Ty, Su, Sy, Rdy, and Rdu.
- Plot Type:** Two radio buttons: ☐ I/O Pair Magnitude and ☒ Singular Value.
- Input/Output:** Input Number: and Output Number: .
- Singular Value Options:** ☒ Show Minimum Singular Value and ☐ Show Maximum Singular Value.
- Axis Labels:**
 - Plot(s) Title:
 - Frequency Axis Label:
 - Magnitude Axis Label:
- Buttons:** Create Window, Close, and Help.

Figure 5 Plot Creation Dialog Box

3.4.2 Destroy Window

When this option is selected, the user will be asked to delete one window by clicking on it. Hitting any key at this moment cancels the window destruction procedure. Windows can also be destroyed by means of the window manager.

3.4.3 Window Size

Selection of default window size to be used at creation of all subsequent windows.

3.4.4 Clear Windows

Clear all windows while retaining information about desired contents.

3.4.5 Destroy All Windows

All plot windows are eliminated.

3.5 HELP Menu

Online help is not available in this version of *MADCADS*.

3.6 MADCADS Plots

One of the important features of *MADCADS* is the ability to monitor the progress of the search algorithm by means of graphics. Section 3.4.1 describes the plot creation dialog, the variables that can be plotted, and the types of plots that can be obtained.

This section provides some additional brief details about the plots.

First of all, hard copies can be obtained by using any screen dump utility. High resolution hard copies, e.g., in the Postscript language, are not available in this version of *MADCADS*.

In all plot windows, red lines are used to plot results of the current iteration. Results from the first iteration for which the window existed are always plotted with blue lines. Constraints (if defined) are shown in orange.

Singular value and I/O pair magnitude frequency responses are always plotted in decibels and the frequencies are always in Hertz.

3.7 File Formats and Organization

As mentioned above, *MADCADS* uses a setup file as a means for specifying a particular design problem. The setup file contains a list of the files that are necessary for the problem at hand. The **order** of the filenames is important and will be discussed shortly. When a setup is loaded, *MADCADS* opens each listed file and reads the appropriate information. *MADCADS* then opens an output file for storage of the new controller. The filename for the new controller is identical to that of the initial controller except *.out* is appended to the end. Hence if the original stabilizing controller is contained in a file called *my.controller* then the redesigned controller will be stored in a file called *my.controller.out*. When a setup is saved, a new setup file is created (with extension *.set*) that contains a list of the files used for the current problem. Since the only items that change during the design process are the controller and iteration parameters, *MADCADS* need only create new files for those two items and it names them in a special way. The controller is stored in a file whose name is that of the setup file with *.g3.ss* (or *.g11.ss* for feedback controller design) appended to it. Similarly, the iteration parameters are stored in a file whose name is also identical to the setup filename with *.iter.params* appended to it. Hence if the setup file to be saved is called *my.problem.set* then the controller (assuming forward path controller design) would be stored in a file called *my.problem.set.g3.ss* and the current iteration parameters would be stored in a file called *my.problem.set.iter.params*. While this method might seem cumbersome, it allows the user to take a "snapshot" of the design process at a particular point in time while still giving him/her the ability to look at the controller or iteration parameters in a format that is identical to that which was used to specify the problem.

IMPORTANT: When a setup file is created during the design process, the original controller output file is not affected in any way and is still used by *MADCADS* just as if no new setup file was created. In order to continue the design with new filenames, the user must load a new setup file (one that contains different filenames).

This section will now continue with a detailed description of the contents of the required files. Each file must be stored in an ASCII format where all values are considered to be double precision unless otherwise stated.

3.7.1 *The setup (.set) file*

The setup file contains an **ordered** list filenames of those files required by *MADCADS*. The format is as follows:

Block 1 filename
Block 2 filename
Block 3 filename
Block 4 filename
Block 5 filename
Block 6 filename
Block 7 filename
Block 8 filename
Block 9 filename
Block 10 filename
Block 11 filename
Iteration Parameters filename
Frequency Data filename
Singular Value Constraint filename
I/O Pair Magnitude Constraint filename
Controller Pole Damping Ratio Constraint filename
Controller Zero Damping Ratio Constraint filename
Controller Pole Damping Factor Constraint filename

IMPORTANT: All input files must exist in the current directory (the directory from which *MADCADS* is executed).

Each filename must conform to the UNIX standard file naming convention and should be located on a line of its own. A filename must be included for items that are to be omitted from the design (ie: suppose Block 1 is not necessary), but the name should be that of a file that **does not exist** in the current working directory.

IMPORTANT: Certain files cannot be omitted from the design process. These files are:

- Plant Block 5
- Controller File (either Block 3 or Block 11 -- NOT BOTH)
- Frequency Data File
- Iteration Parameters File
- At Least ONE Constraint File

3.7.2 *The Controller*

Only one controller at a time can be designed by this version of *MADCADS*. Hence only one existing controller file (Block 3 or Block 11) can be specified in the setup list. The format of the controller file is that of a special state-variable

representation, and **must** initially stabilize the system. This representation comes from a parameterization of the controller called the *state feedback parameterization* and requires the storage of 3 integers and 5 double precision matrices. Specifically, the state-variable representation (A, B, C, D) must have the form $(A_F + BF, B, C, D)$ where $A = A_F + BF$ and $F \in \mathbb{R}^{n \times p}$ is chosen such that the pair (A_F, F) have desirable characteristics. The data file itself must contain the following (in an ASCII format):

```

controller order (integer)
number of controller outputs (integer)
number of controller inputs (integer)
controller matrix A stored columnwise - one element per line
controller matrix B stored columnwise - one element per line
controller matrix C stored columnwise - one element per line
controller matrix D stored columnwise - one element per line
controller parameterization matrix F stored columnwise - one element per line

```

3.7.3 Frequency Response Matrix Blocks

All system blocks except Block 3 and Block 11 must be specified as frequency response data. Since we are dealing with the multivariable case, we must have frequency response matrix data. Thus we must have one response matrix for each frequency point stored in the frequency data file (Section 3.7.5). The file format is as follows (in ASCII):

```

number of block outputs (integer)
number of block inputs (integer)
frequency response matrix for first frequency point stored columnwise
    with one complex value per line -- x y -- where x is real part, y is imaginary part
frequency response matrix for second frequency point stored columnwise
    with one complex value per line -- x y -- where x is real part, y is imaginary part

etc... until entire response is specified

```

3.7.4 Iteration Parameters

This file contains the iteration parameters as follows:

```

step length reduction factor
step length increase factor
initial step length
Armijo factor
initial epsilon boundary

```

3.7.5 Frequency Data

This file contains the sampling period (in seconds) followed by a list of discrete frequency points (in Hz) at which the response data for the problem is defined.

3.7.6 Singular Value Constraint Data

This file contains the information necessary for *MADCADS* to build the singular value constraint curves. The first value in the file is an integer that defines how many singular value constraints exist. For each constraint, the user must supply the following information:

- constraint_name (text)
- constraint_direction (text)
- number of defining points (integer)
- frequency point/singular value (in dB) pair for each defining point

The text values come from:

constraint_name:

This is the name of the frequency response matrix being constrained. It **must be identical** (except for case) to that found in the plot definition dialog. Hence it must be one of:

tun, tur, tuw, tyn, tyr, tyw, tzn, tzt, tzu, tu, ty, su, sy, rdu, rdy

constraint_direction:

This is a string of text that indicates whether the constraint curve defines an upper or lower constraint. The valid values are:

up, low

The defining points must be in order of ascending frequency and must completely cover the entire frequency range defined in the frequency data file. *MADCADS* will interpolate constraint values in a linear fashion for all frequencies between the defining points.

The following is an example of a singular value constraint file that includes an **upper** constraint on *tuw* and a **lower** constraint on *rdy*. The frequency range is assumed to cover 0.01 Hz to 25.0 Hz., and each constraint will have 3 points of definition. Specifically, the constraint on *tuw* will require that all singular values are below 0 dB at 0.01 Hz, below -5.0 dB at 1 Hz and below -30.0 dB at 25.0 Hz. Similarly, we will require that all singular values of *rdy* are above -15 dB at 0.01 Hz, above -3 dB at 10 Hz and above 2 dB at 25 Hz. The file would look as follows:

```

2
tuw up 3
1e-02 0.0
1.0 -5.0
25.0 -30.0
rdy low 3
1e-02 -15.0
10.0 -3.0
25.0 2.0

```

IMPORTANT: Only one set of constraints can be made for a particular response matrix.

3.7.7 I/O Pair Magnitude Constraint Data

This file contains the information necessary for *MADCADS* to build the I/O pair magnitude constraint curves. The first value in the file is an integer that defines how many I/O pair magnitude constraints exist. For each constraint, the user must supply the following information:

```

constraint_name (text)
constraint_direction (text)
output index (transfer function matrix row) (integer)
input index (transfer function matrix column) (integer)
number of defining points (integer)
frequency point/magnitude value (in dB) pair for each defining point

```

The text values come from:

constraint_name:

This is the name of the frequency response matrix being constrained. It **must be identical** (except for case) to that found in the plot definition dialog. Hence it must be one of:

tun, tur, tuw, tyn, tyr, tyw, tzn, tzt, tzu, tu, ty, su, sy, rdu, rdy

constraint_direction:

This is a string of text that indicates whether the constraint curve defines an upper or lower constraint. The valid values are:

up, low

The defining points must be in order of ascending frequency and must completely cover the entire frequency range defined in the frequency data file. *MADCADS* will interpolate constraint values in a linear fashion for all frequencies between the defining points.

The following is an example of an I/O pair magnitude constraint file that includes a an **upper** constraint on the (3,4) element of *tuw* and a **lower** constraint on the (2,1) element of *tyn*. The frequency range is assumed to cover 0.01 Hz to 25.0 Hz., and each constraint will have 3 points of definition. Specifically, the constraint on *tuw* will require that the magnitude of the frequency response from the 4th input the 3rd output of *tuw* be below 0 dB at 0.01 Hz, below -5.0 dB at 1 Hz and below -30.0 dB at 25.0 Hz. Similarly, we will require that the magnitude of the frequency response from the 1st input the 2nd output of *tyn* be above -15 dB at 0.01 Hz, above -3 dB at 10 Hz and above 2 dB at 25 Hz. The file would look as follows:

```
2
tuw up 3 4 3
1e-02 0.0
1.0 -5.0
25.0 -30.0
rdy low 2 1 3
1e-02 -15.0
10.0 -3.0
25.0 2.0
```

IMPORTANT: Only one set of constraints can be made for a particular response matrix.

3.7.8 *Controller Pole Damping Ratio Data*

This file contains one value that specifies the constraint on the damping ratios of the controller poles.

3.7.9 *Controller Zero Damping Ratio Data*

This file contains one value that specifies the constraint on the damping ratios of the controller zeros.

3.7.10 *Controller Pole Damping Factor Data*

This file contains one value that specifies the constraint on the damping factors of the controller poles.

4.0 References and Additional Information

- [1] J. R. Mitchell, R. D. Irwin, E. A. Medina, D. A. Allwine, W. G. Frazier, and M. A. Duncan (1995), *Computerized Design of Controllers Using Data Models*, Final Report, NASA Grant NAG8-217, Department of Electrical and Computer Engineering, Ohio University, Athens, Ohio 45701, July 1995.
- [2] Open Software Foundation (1990), *OSF/MotifTM User's Guide*, Revision 1.0, Prentice Hall, Englewood Cliffs, New Jersey.
- [3] D. Heller (1991), *Motif Programming Manual*, O'Reilly & Associates, Inc., Sebastopol, CA, 1991.
- [4] Andrew S. Glassner, Editor (1990), *Graphics Gems*, Academic Press, Cambridge, MA, 1990.

NOTE: Reference [1] contains an in-depth discussion of the theoretical background for *MADCADS* as well as the results of the application of *MADCADS* to several real-world problems.

Notes

The graphical user interface (GUI) for *MADCADS* was written in C working with the X Window System, release X11R5, using the Motif Toolkit, release 1.2.4. Some of the GUI modules are based on examples included in [3]. The following is the Copyright notice that accompanies all programs in reference [3].

```
/* Written by Dan Heller. Copyright 1991, O'Reilly & Associates.  
 * This program is freely distributable without licensing fees and  
 * is provided without guarantee or warranty expressed or implied.  
 * This program is -not- in the public domain. */
```

Loose labels for plots were implemented by using a C routine written by Paul Heckbert, which can be found in [4]. The following note accompanies the software of reference [4].

The authors and the publisher hold no copyright restrictions on any of these files; this source code is public domain, and is freely available to the entire computer graphics community for study, use, and modification. We do request that the comment at the top of each file, identifying the original author and its original publication in the book *Graphics Gems*, be retained in all programs that use these files.

The matrix widget *XbaeMatrix*, version 3.8, was used for dialog boxes in which matrices would provide ease of data input. The Copyright status of the *XbaeMatrix* package is given by the following note. The release of *XbaeMatrix* used included third party modifications, as written at the end of the note.

```
/*  
 * Copyright(c) 1992 Bell Communications Research, Inc. (Bellcore)  
 * All rights reserved  
 * Permission to use, copy, modify and distribute this material for any purpose and without fee is  
 * hereby granted, provided that the above copyright notice and this permission notice appear in  
 * all copies, and that the name of Bellcore not be used in advertising or publicity pertaining to  
 * this material without the specific, prior written permission of an authorized representative of  
 * Bellcore.  
 *  
 * BELLCORE MAKES NO REPRESENTATIONS AND EXTENDS NO WARRANTIES, EXPRESS OR  
 * IMPLIED, WITH RESPECT TO THE SOFTWARE, INCLUDING, BUT NOT LIMITED TO, THE  
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR ANY PARTICULAR  
 * PURPOSE, AND THE WARRANTY AGAINST INFRINGEMENT OF PATENTS OR OTHER  
 * INTELLECTUAL PROPERTY RIGHTS. THE SOFTWARE IS PROVIDED "AS IS", AND IN NO  
 * EVENT SHALL BELLCORE OR ANY OF ITS AFFILIATES BE LIABLE FOR ANY DAMAGES,  
 * INCLUDING ANY LOST PROFITS OR OTHER INCIDENTAL OR CONSEQUENTIAL DAMAGES  
 * RELATING TO THE SOFTWARE.  
 *  
 * MatrixWidget Author: Andrew Wason, Bellcore, aw@bae.bellcore.com  
 *  
 * Modification History: David Boerschlein, dpb@air16.larc.nasa.gov  
 * Lockheed Engineering & Sciences Company,  
 * Under contract to NASA LaRC:
```